DESARROLLO DE UNA APLICACIÓN WEB PARA LA UNIDAD DE INFORMACIÓN Y FORTALECIMIENTO DE CAPACIDADES DEL CIAT

LIDA DEL ROSARIO LA HOZ GÓMEZ

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE CIENCIAS DE LA INFORMACIÓN
PROGRAMA DE INGENIERÍA INFORMÁTICA
SANTIAGO DE CALI
2006

DESARROLLO DE UNA APLICACIÓN WEB PARA LA UNIDAD DE INFORMACIÓN Y FORTALECIMIENTO DE CAPACIDADES DEL CIAT

LIDA DEL ROSARIO DE LA HOZ GÓMEZ

Pasantía para optar el título de
Ingeniero en Informática

Directora
MARY ELIZABETH RAMIREZ CANO
Ingeniera de Sistemas

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE CIENCIAS DE LA INFORMACIÓN
PROGRAMA DE INGENIERÍA INFORMÁTICA
SANTIAGO DE CALI
2006

Nota de Aceptación:

Aprobado por el Comité de Grado en cumplimiento de los requisitos exigidos por la Universidad Autónoma de Occidente para optar por el título de Ingeniero en Informática

Ingeniera MARY ELIZABETH RAMIREZ
Directora Académica del Proyecto

Santiago de Cali, Mayo 25 de 2006

AGRADECIMIENTOS

Agradezco a mis padres Narciso De la Hoz y Edith Gómez y a mis hermanos Leda, Narciso, Liliana y Diana, quienes me brindaron su apoyo desde que decidí empezar una nueva carrera. A las Ingenieras Lyda Peña y Mary Ramírez por todo lo que me enseñaron durante mi paso por la Universidad. Al Doctor Jaime Campo, porque gracias a él llegué al CIAT. A Julián Micolta quien me ayudó durante el desarrollo del proyecto y a mis jefes en el CIAT los Ingenieros Carlos Meneses, Luis Fernando Cruz y Doryan Colunge.

CONTENIDO

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

Pág.

LISTA DE ANEXOS

Pág.

GLOSARIO

ARQUITECTURA DE TRES CAPAS: estilo de programación por capas en el que el objetivo principal es separar la lógica de negocio de la de diseño. Las capas son la de presentación, la de negocio y la de datos.

DBForms: framework que le permite al desarrollador construir aplicaciones weben corto tiempo y está basado en las especificaciones Java Servlets 2.3 y Java Server Pages 1.2.

EXTREME PROGRAMMING: es una aproximación a la ingeniería de software formulada por Kent Beck, caracterizada por el desarrollo iterativo e incremental, la programación por parejas, la frecuente interacción del equipo de programación con el cliente, la simplicidad del código y la realización de pruebas unitarias durante todo el desarrollo del software.

FRAMEWORK: estructura de soporte definida, donde otro proyecto de software puede ser desarrollado. Incluye soporte de programas, librerías, lenguaje de script, y tags para ayudar a desarrollar y unir los componentes de un proyecto.

IDE: entorno o ambiente integrado de desarrollo por sus siglas en inglés (Integrated Development Environment) es un programa compuesto por un conjunto de herramientas para el programador.

InforCAP: Unidad de Información y Fortalecimiento de Capacidades, Unidad del CIAT que se encarga de la promoción y divulgación de cursos, eventos y seminarios.

JAR: o Java Archives, es un formato desarrollado por Sun Microsystems que permite agrupar las clases diseñadas en lenguaje Java, otorgando un nivel de compresión; permite empaquetar varios archivos en uno solo.

MODELO VISTA CONTROLADOR: patrón de diseño de software que separa los datos de la aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos. Se ve frecuentemente empleado en aplicaciones web.

OUTPOSTED: empleado del CIAT que trabaja fuera de la sede de Palmira – Valle.

PROCESO UNIFICADO: proceso de desarrollo de software que junto con el Lenguaje Unificado de Modelado constituye la metodología estándar mas empleada para el análisis, implementación y documentación de sistemas orientados a objetos.

TAG: se refiere a etiquetas o marcas en los lenguajes de programación.

UML: lenguaje Unificado de Modelado (Unified Modeling Lenguaje) es el lenguaje de modelado de sistemas de software más empleado en la actualidad, está apoyado por el OMG, Object Management Group, y se emplea para visualizar, especificar, construir y documentar un sistema de software.

WAR: Web Archive, es un tipo de archivo jar que puede contener varios tipos de fuentes e imágenes como *.java, *.jsp, *.xml, *.css, *.ejb, *.gif, *.html, *.png, *.sql, *.xsl. Toda aplicación en Tomcat se encuentra agrupada en este tipo de archivo. La estructura de un archivo war es la siguiente:

- / *.html *.jsp *.css : Este directorio base contiene los elementos como documentos HTML , JSP's , CSS("Cascading Style Sheets") entre otros.
- /WEB-INF/web.xml : Contiene elementos de seguridad de la aplicación así como detalles sobre los Servlets que serán utilizados dentro de la misma.
- /WEB-INF/classes/ : Contiene las clases Java adicionales a las del JDK que serán empleadas en los JSP's y Servlets
- /WEB-INF/lib/ : Contiene los JAR's que serán utilizados en la aplicación.

RESUMEN

Para el desarrollo de la aplicación Web para el manejo de la información de los cursos y eventos ofrecidos por la Unidad de Información y Fortalecimiento de Capacidades del CIAT en sus oficinas en Palmira y las outposted se empleó la metodología de desarrollo propia de la Unidad de Sistemas de Información del CIAT, que agrupa conceptos de otras metodologías como la Extreme Programing, Desarrollo de Procesos, y la Teoría de Gerencia de Proyectos, con la ayuda del lenguaje de modelado unificado UML.

El levantamiento de requerimientos se realizó mediante entrevistas a los usuarios y con la aplicación que se encontraba en funcionamiento en ese momento. Posteriormente se realizaron los diagramas de casos de uso, se realizaron las modificaciones pertinentes a la base de datos y se seleccionó el framework DBForms para el desarrollo del proyecto, el cual disminuye el tiempo de codificación pues trae características predeterminadas, y como entorno de desarrollo se seleccionó Eclipse con el plugin MyEclipse que le añade características que permiten la creación de aplicaciones Web.

Durante todo el proceso de desarrollo se realizaron pruebas y una vez finalizada esta etapa se realizaron las pruebas formales a la aplicación que se encontraba funcionando en un servidor de aplicaciones de prueba, posteriormente se capacitó a los usuarios y finalmente fue instalada en el servidor de producción y se dejó a disposición de los usuarios de CIAT Palmira.

Al aplicar la metodología de desarrollo se observó que el proyecto cumplió con las metas establecidas y se logró la satisfacción de los usuarios de la aplicación.

INTRODUCCIÓN

El Centro Internacional de Agricultura Tropical, CIAT, es una institución sin ánimo de lucro dedicada a la investigación social orientada hacia la reducción del hambre y la pobreza y hacia la preservación de los recursos naturales en los países en desarrollo. El CIAT es uno de los 15 centros financiados principalmente por 58 países, fundaciones privadas y organizaciones internacionales que constituyen el Grupo Consultivo para la Investigación Agrícola Internacional CGIAR, por su sigla en inglés.

Dentro de la estructura organizacional del CIAT se encuentran diferentes áreas como las de Dirección, Administración y Coordinación Regional, Agrobiodiversidad y Genética, Ecología y manejo de Plagas y Enfermedades, Ecología y Mejoramiento de Suelos, Análisis de Información Espacial, Análisis Socioeconómico, además del área de Apoyo a la Investigación, entre las cuales se encuentran la Unidad de Comunicaciones, la Unidad de Sistemas de Información y la Unidad de Información y Fortalecimiento de Capacidades.

La Unidad de Información y Fortalecimiento de Capacidades, InforCAP, tiene como misión “el fortalecimiento de la capacidad científica a través de entrenamientos grupales especializados y multidisciplinarios, de capacitación individualizada y tesis de maestría y/o doctorados realizados en el CIAT”; como herramienta de apoyo para el cumplimiento de su misión, la Unidad posee una base de datos Oracle en la cual existe información que data del año 1970, sobre los usuarios que reciben capacitación, el tipo de entrenamiento que reciben, las entidades que los patrocinan, el origen de los fondos, las instituciones de las que provienen los usuarios, las disciplinas en las que se clasifican los cursos ofrecidos, y las personas encargadas de supervisar cada uno de los cursos.

Para el manejo de la base de datos mencionada se desarrolló en Forms de Oracle una aplicación cliente-servidor, puesto que la información se administraba de manera centralizada en el CIAT. Con la posterior oferta de cursos de capacitación en Asia, África y Centro América, surgió la necesidad de que los outposted (personal del CIAT que trabaja en el exterior) pudieran ingresar la información que sobre cursos y entrenamientos se genera en su sitio de trabajo, lo cual no puede ser cubierto por la aplicación actual, razón que ha motivado la búsqueda de una aplicación de bajo costo que brinde el acceso desde cualquier sitio del mundo, además de conservar los requerimientos básicos de funcionamiento de la plataforma actual.

1. PLANTEAMIENTO DEL PROBLEMA

Durante los últimos años la aplicación de Capacitación venía cumpliendo con el objetivo para el que fue creada: la administración de la información generada en la Unidad de Información y Fortalecimiento de Capacidades: ingreso, modificación y consulta de usuarios, entrenamientos, clases, disciplinas del conocimiento, supervisores de cursos, presupuestos manejados y eventos realizados.

Dado que en los últimos meses se ha presentado la necesidad de ingresar información no sólo desde el CIAT, sino desde cualquier estación outposted, y de tener un manejo integral de la información, la Unidad de Información y Fortalecimiento de Capacidades plantea el interrogante de *Cómo permitir el acceso a la base de datos existente de manera segura desde cualquiera de las oficinas que el CIAT tiene en Asia, África y Centro América, de tal manera que esos datos se transformen en información útil y confiable para reportarla de manera oportuna a todas las Unidades de Trabajo del CIAT que la requieran?*

2. MARCO TEÓRICO

Para dar solución a las necesidades de administración de información en la Unidad de Información y Fortalecimiento de Capacidades, la Unidad de Sistemas de Información del CIAT, USI, desarrolló aproximadamente en 1996 en Forms de Oracle y bajo el esquema cliente – servidor el Sistema de Información de Capacitación y Comunicaciones del CIAT, ya que se requería de una aplicación que estuviera acorde a las necesidades especificas del Área, y consta de los siguientes módulos:

- Biblioteca
- Capacitación
- Conferencias
- Inventarios
- Pregrados y Acuerdos
- Publicaciones
- Salas

Una base de datos única elaborada en Oracle, almacena la información que es ingresada en cada uno de los módulos, por lo tanto algunas de las tablas contienen información compartida, como la de usuarios, disciplinas, presupuestos, países, instituciones, supervisores, referencias, dominios, sectores, scope.

El módulo de Capacitación consta de las siguientes opciones: Usuarios, Instituciones, Países, Disciplinas, Supervisores, Clases, y Presupuestos, las cuales permiten realizar funciones básicas como inserción, edición y consulta, además cuenta con un módulo para la elaboración e impresión de reportes predeterminados.

Para el año 2000, la Unidad de Sistemas de Información realizó una actualización de la aplicación que es la que actualmente se encuentra en producción.

3. ANTECEDENTES

La propuesta de migrar la Aplicación de Capacitación de un esquema cliente-servidor a uno con arquitectura Web se originó en la Unidad de Sistemas de Información del CIAT, en respuesta a la solicitud planteada por la Unidad de Información y Fortalecimiento de Capacidades debido al surgimiento de un nuevo requerimiento que debía cumplir su actual Sistema de Información: el de permitir el ingreso a la base de datos de información sobre capacitaciones, usuarios y eventos realizados por el CIAT en sus sedes de África, Asia y Centro América directamente por el personal que se encuentra laborando en esas oficinas.

Actualmente la labor de alimentación, y actualización de la información que se genera sobre cursos y capacitaciones ofrecidos por el CIAT en sus diferentes sedes, la realiza el personal de CIAT Palmira, los locales con los registros que se obtienen de los participantes y los del extranjero con documentos en papel que son enviados por correo por los encargados de las estaciones outposted; esta ultima presenta el inconveniente de que en ocasiones los datos llegan incompletos, mal diligenciados o con días de retraso, lo que se traduce en falta de oportunidad para la realización de los reportes que las diferentes áreas del CIAT solicitan.

Por otro lado, la herramienta de generación de reportes que posee el Sistema de Información actual, solo permite la elaboración de reportes predeterminados, estáticos y con el transcurrir de los años las necesidades de información de los programas han cambiado, por lo cual se necesita la implementación de una solución que permita que éstos sean generados de manera personalizada según las solicitudes que requieran realizadas los programas del CIAT.

Lo que se persigue con el nuevo Sistema de Información, es que el personal de cada sede se encargue de administrar la información de los eventos que se realicen en su área de influencia, que se cuente con medidas de seguridad que impidan el acceso no autorizado a la aplicación, que el sistema permita búsquedas precisas, y que la generación de reportes sea personalizada.

Dado lo anterior, el CIAT ha solicitado a la Universidad Autónoma de Occidente la colaboración para el desarrollo de este proyecto y se ha dado respuesta a través de una pasantía como opción de grado.

4. OBJETIVOS

4.1 OBJETIVO GENERAL

Desarrollar una aplicación Web que permita al personal del área de Comunicación y Capacitación del CIAT y al personal outposted de África y Asia, el ingreso de los datos de las capacitaciones ofrecidas en el CIAT y en los demás centros de Investigación con el fin de tener información oportuna para reportar a las áreas que lo necesiten.

4.2 OBJETIVOS ESPECÍFICOS

- Levantamiento de requerimientos, elaboración de las historias del cliente y de la matriz de requerimientos.
- Realizar el Análisis de requerimientos.
- Elaborar los diagramas de secuencia y clases y realizar las modificaciones al Modelo Entidad Relación existente.
- Realizar la codificación de la aplicación.
- Realizar la matriz de casos de prueba.
- Realizar las pruebas.
- Implementar la aplicación en el ambiente de producción.

5. JUSTIFICACIÓN

Con el desarrollo de la aplicación Web descrita la Unidad de Información y Fortalecimiento de Capacidades obtendrá los siguientes beneficios:

- Fácil acceso a la información, gracias a que los datos sobre cursos, seminarios y entrenamientos podrá ser ingresada y consultada en la base de datos directamente desde cada una de las sedes que posee el CIAT en Centro América, Asia y África.

- Oportunidad en la elaboración de reportes, debido a que el ingreso de la información a la base de datos no contará con intermediarios, la información siempre estará actualizada, permitiendo generar los reportes que los diferentes programas del CIAT soliciten.

- Disminución de tiempo, puesto que se evitará el envío de información por correo desde las oficinas outposted para que sea ingresada a la base de datos en la sede de CIAT Palmira.

- Flexibilidad en la elaboración de reportes, pues podrán ser personalizados y generados en diferentes formatos como PDF y Excel.

6. METODOLOGÍA

Con el fin de cumplir con los objetivos del proyecto se empleará la metodología de elaboración de proyectos de la Unidad de Sistemas de Información del CIAT, la cual se basa en conceptos de Extreme Programming, en ideas de otras metodologías como el Proceso Unificado, la teoría Gerencia de Proyectos y experiencias CIAT en las cinco etapas de un proyecto de desarrollo, las cuales son: Planeación, Análisis, Desarrollo o Codificación, Pruebas y Producción, y apoyado en el Lenguaje de Modelado Unificado UML. A continuación se explican en detalle:

PLANEACIÓN

- Obtención y Análisis de requerimientos:
    - Reuniones/Entrevistas con los usuarios.
    - Conocimiento del sistema de información de capacitación.
    - Elaboración de las historias del cliente.
    - Elaboración de la Matriz de requerimientos.

DISEÑO

- Elaborar los cambios en el Modelo Entidad Relación.

- Elaboración de los diagramas de casos de uso, de clases, de secuencia y de componentes.

- Elaboración de la matriz de requerimientos de prueba.

CODIFICACIÓN

- Realizar el desarrollo de las funcionalidades del sistema con las herramientas que para tal fin cuenta la Unidad de Sistemas de Información.

PRUEBAS

- Realizar pruebas funcionales y de sistema.

IMPLEMENTACIÓN

- Configuración/Instalación del aplicativo.
- Realización del acta de aceptación con el cliente.

CAPACITACIÓN DE USUARIOS FINALES

DESARROLLO DEL PROYECTO

7. ETAPA DE ANÁLISIS

7.1 LEVANTAMIENTO DE REQUERIMIENTOS

Con el fin de mejorar el servicio al personal de capacitación y outposted en el manejo de las capacitaciones, se requiere mejorar el sistema actual para que permita el acceso remoto a la misma, con el objetivo de lograr que la información pueda ser ingresada desde Asia y África. Por lo anterior, se revisaron las necesidades de este personal y se encontraron los requerimientos para modificar y/o actualizar el sistema actual, que se describen a continuación:

7.1.1 Requerimientos generales

R-G1 La aplicación existente bajo el esquema cliente/servidor debe ser migrada a una aplicación web.

R-G2 Los datos existentes en la base de datos actual deben ser conservados.

R-G3 El sistema debe permitir la generación de reportes de las capacitaciones ofrecidas por continente.

R-G4 El sistema debe enviar por correo electrónico una notificación al usuario y al administrador de la aplicación cuando éste haya ingresado al sistema sus datos.

R-G5 Para permitir flexibilidad total sobre la elaboración de los reportes, se empleará la herramienta Discoverer de Oracle.

R-G6 La interfaz gráfica de la aplicación debe estar de acuerdo a la de la Intranet del CIAT, Arconet.

7.1.2 Users – Usuarios: estos requerimientos se refieren al ingreso, consulta y modificación de los usuarios que reciben capacitación o que participan en eventos en el CIAT o en sus oficinas outposted.

R-U1. El sistema debe permitir al administrador de la aplicación y al usuario outposted el ingreso de nuevos usuarios (Users) en la base de datos.

R-U2. El sistema debe permitir al administrador de la aplicación la modificación de los datos de los usuarios existentes en la base de datos, que hayan sido creados o no por él.

R-U3. El sistema debe permitir al usuario outposted la modificación de los usuarios que él haya creado.

R-U4. El sistema debe permitir al administrador de la aplicación y al usuario outposted consultar los datos de los usuarios existentes en la base de datos.

R-U5. El sistema debe permitir al administrador de la aplicación y al usuario outposted la búsqueda de usuArios por nombre, apellido, combinación de ellos y por código.

7.1.3 Training – Entrenamiento: estos requerimientos se refieren a los entrenamientos realizados por un usuario en particular en el CIAT o en alguna de las oficinas outposted.

R-E1. El sistema debe permitir al administrador de la aplicación y al usuario outposted el ingreso de los datos de los entrenamientos (Trainings) recibidos por los usuarios.

R-E2. El sistema debe permitir al administrador de la aplicación la modificación de los datos de los entrenamientos recibidos por los usuarios, que hayan sido creados o no por él.

R-E3. El sistema debe permitir al usuario outposted la modificación de los entrenamientos que él haya creado.

R-E4. El sistema debe permitir al administrador de la aplicación y al usuario outposted consultar los datos de los entrenamientos recibidos por los usuarios.

R-E5. El sistema debe permitir al usuario administrador y al usuario outposted realizar búsquedas de entrenamientos realizados por código de usuario.

7.1.4 Events – Eventos

R-EV1 El sistema deberá permitir al usuario outposted y al adminitrador la creación de nuevos eventos.

R-EV2 El sistema deberá permitir al administrador de la aplicación la modificación de todos los eventos que se encuentren almacenados en la base de datos.

R-EV3 El sistema deberá permitir al usuario outposted la modificación de los usuarios que el haya creado.

7.1.5 Institutions - Instituciones

R-I1. El sistema debe permitir al administrador de la aplicación y al usuario outposted el ingreso de nuevas instituciones.

R-I2. El sistema debe permitir al administrador de la aplicación la modificación de los datos de las instituciones que hayan sido creados o no por él.

R-I3. El sistema debe permitir al usuario outposted la modificación de las instituciones que hayan sido creadas por él.

R-I4. El sistema debe permitir al administrador de la aplicación y al usuario outposted consultar los datos de las instituciones existentes en la base de datos.

R-I5. El sistema debe permitir al administrador de la aplicación y al usuario outposted la búsqueda de instituciones por su nombre.

7.1.6 AltInst

R-AI R-AI1. El sistema debe permitir al administrador de la aplicación la búsqueda de Instituciones que ya están asociadas a un training.

R-AI R-AI2. El sistema debe permitir al administrador de la aplicación adicionar nuevas Instituciones asociadas a un país para que puedan ser relacionadas al entrenamiento de un usuario.

7.1.7 Countries - Países

R-PA1 El sistema debe permitir al administrador de la aplicación el ingreso de nuevos países, asociándolos a un continente y asociándoles un estado (activo o inactivo).

R-PA2 El sistema debe permitir al administrador de la aplicación la modificación de los datos de los países.

R-PA3 El sistema debe permitir al administrador de la aplicación consultar los datos de los países existentes en la base de datos.

R-PA4 El sistema debe permitir al administrador de la aplicación la búsqueda de países por nombre.

R-PA5 El sistema deberá asignar un estado activo o inactivo a los países, como en el caso anterior se necesita mantener el registro histórico, pero solo deben asociarse usuarios o entrenamientos con países que existan en la actual distribución política, teniendo en cuenta los cambios que han surgido en ella en los últimos veinte años, en los cuales se ha guardado información en la base de datos.

7.1.8 Continents - Continentes

R-C1. El sistema debe permitir al administrador de la aplicación el ingreso de continentes.

R-C2. El sistema debe permitir al administrador de la aplicación la modificación de los datos de los continentes.

R-C3. El sistema debe permitir al administrador de la aplicación consultar los datos de los continentes existentes en la base de datos.

7.1.9 Disciplines - Disciplinas

R-D1. El sistema debe permitir al administrador de la aplicación el ingreso de nuevas disciplinas.

R-D2. El sistema debe permitir al administrador de la aplicación la modificación de los datos de las disciplinas.

R-D3. El sistema debe permitir al administrador de la aplicación consultar los datos de las disciplinas existentes en la base de datos.

R-D4. El sistema debe permitir al administrador de la aplicación asignar un estado a las disciplinas, activo o inactivo.

R-D5. El sistema deberá permitir al administrador la búsqueda de disciplinas por nombre.

R-D6. El sistema deberá permitir asignar un estado activo o inactivo a las disciplinas, pues algunas de ellas se han modificado o agrupado y se necesita conservar la información histórica, pero para los usuarios sólo se deben mostrar las que están activas y no permitir que se asignen las que ya no existen, han cambiado de nombre o se han agrupado.

7.1.10 Classes - Clases

R-CL1. El sistema debe permitir al administrador de la aplicación el ingreso de nuevas clases de capacitación.

R-CL2. El sistema debe permitir al administrador de la aplicación la modificación de las clases de capacitación existentes.

R-CL3. El sistema debe permitir al administrador de la aplicación consultar los datos de las clases de capacitación existentes en la base de datos.

R-CL4. El sistema debe permitir al administrador de la aplicación la búsqueda de las clases de capacitación existentes.

R-CL5. El sistema debe permitir al administrador de la aplicación asociar un estado a las clases, activo o inactivo.

R-CL6. El sistema debe permitir al administrador de la aplicación la búsqueda de clases por nombre.

R-CL7. El sistema deberá permitir asignar un estado activo o inactivo a las clases, pues algunas de ellas se han modificado o agrupado y se necesita conservar la información histórica, pero para los usuarios sólo se deben mostrar las que están activas y no permitir que se asignen las que ya no existen, han cambiado de nombre o se han agrupado.

7.1.11 Budget - Presupuesto

R-PR1. El sistema debe permitir al administrador de la aplicación el ingreso de nuevos presupuestos.

R-PR2. El sistema debe permitir al administrador de la aplicación la modificación de los datos de los presupuestos.

R-PR3. El sistema debe permitir al administrador de la aplicación consultar los datos de los presupuestos existentes en la base de datos.

7.1.12 Supervisors - Supervisores

R-S1. El sistema debe permitir al administrador de la aplicación el ingreso de nuevos supervisores.

R-S2. El sistema debe permitir al administrador de la aplicación la modificación de los datos de los supervisores.

R-S3. El sistema debe permitir al administrador de la aplicación consultar los datos de los supervisores existentes en la base de datos.

R-S4. El sistema debe permitir al administrador de la aplicación la búsqueda de Supervisores por nombre.

7.1.13 Administración: estos requerimientos hacen referencia a la administración de los usuarios que van a manipular información en la aplicación (usuarios administradores/outposted).

R-A1. El sistema debe contar con dos tipos de usuarios:

- Administrador: usuario que tendrá privilegios para administrar/modificar el sistema: Users, Training, Institutions, Disciplines, Classes, Countries, Continents, Supervisors, Budgets que haya o no creado, y realizar reportes sobre la información almacenada en la base de datos.

- Outposted: podrá ingresar y modificar información de Users, Training, Institutions y Eventos que genere cada oficina outposted del CIAT.

R-A2 El sistema debe permitirle al administrador de la aplicación la creación de nuevos usuarios asignándoles código y password.

R-A3 El sistema debe permitir al Administrador de la aplicación asignarles uno de los roles existentes a los usuarios.

R-A4 El sistema debe permitir al administrador de la aplicación la actualización de los datos de los usuarios.

R-A5 El sistema debe permitir al administrador de la aplicación eliminar usuarios existentes de la base de datos.

R-A6 El sistema debe permitirle al administrador de la aplicación consultar los datos de los usuarios existentes en la base de datos.

R-A7 El sistema debe permitir a los usuarios la modificación de sus datos personales.

7.1.14 Secutity

R-SG1 El sistema deberá solicitar password y login al usuario para su ingreso al sistema.

R-SG2 Según los privilegios del usuario registrado, el sistema debe permitir o restringir el acceso a las diferentes opciones del menú.

R-SG3 El sistema debe contar con una opción que permita la salida segura del sistema.

7.1.15 Búsquedas

R-B1 El sistema debe permitirle al administrador de la aplicación la búsqueda de usuarios por nombre y/o apellido.

R-B2 El sistema debe permitirle al adminstrador de la aplicación la búsqueda de usuarios por código.

R-B3 El sistema debe permitirle al administrador de la aplicación realizar búsquedas por instituciones.

7.2 CASOS DE USO

Luego de que los requerimientos fueron analizados, se establecieron los actores y el listado de casos de uso que se muestran a continuación.

7.2.1 Actores

- **AppAdministrator:** Usuario que cuenta con todos los privilegios sobre la aplicación, puede ingresar, modificar, actualizar y consultar información sobre usuarios, entrenamientos, eventos, clases, disciplinas, supervisores, instituciones, presupuestos, países y continentes que hayan sido o no creadas por él.

- **Outposted:** Usuario que tiene privilegios para ingresar y modificar datos sobre usuarios, entrenamientos, eventos e instituciones que él adicione a la base de datos, además puede consultar información sobre los que no ha creado.

La Tabla 1 resume las opciones a las que tiene acceso cada uno de los tipos de usuario de la aplicación.

Tabla 1. Opciones del menú disponibles según el tipo de usuario de la aplicación.

Opción	AppAdministrator	Outposted
Users	+	+
Training	+	+
Institutions	+	+
AltInstitutions	+	+
Event	+	+
Continents	+	-
Countries	+	-
Disciplines	+	-
Supervisor	+	-

Classes	+	-
Budgets	+	-
Administration	+	-

7.2.2 Lista de casos de uso. Del listado de requerimientos se obtuvieron los siguientes casos de uso, agrupados por módulos, según funcionalidad. Los datos de entrada para cada caso de uso se encuentran en el Anexo A.

Módulo Users

- UC_Create_New_User
- UC_Update_User
- UC_Get_User_Details
- UC_Search_User

Módulo Training.

- UC_Create_New_Training
- UC_Update_Training
- UC_Get_Training_Details
- UC_Search_Training

Módulo Events.

- UC_Create_New_Event
- UC_Update_Event
- UC_Get_Event_Details
- UC_Search_Event

Módulo Institutions

- UC_Create_New_Institution
- UC_Update_Institution
- UC_Get_Institution_Details
- UC_Search_Institutions

Módulo AltInstitutions

- UC_Add_New_Institution
- UC_Search_Institutions

Módulo Countries

- UC_Create_New_Countries
- UC_Update_Countries
- UC_Get_Countries_Details
- UC_Search_Country
- UC_Get_Country_List

Módulo Continents

- UC_Create_New_Continents
- UC_Update_Continents
- UC_Get_Continent_List

Módulo Disciplines

- UC_Create_New_Disciplines
- UC_Update_Disciplines
- UC_Search_Disciplines
- UC_Get_Disciplines_List

Módulo Classes

- UC_Create_New_Class
- UC_Update_Class
- UC_Search_Class
- UC_Get_Class_List

Módulo Budgets

- UC_Create_New_Budgets
- UC_Update_Budgets
- UC_Get_Continent_List

Módulo Supervisors

- UC_Create_New_Supervisor
- UC_Update_Supervisors
- UC_Search_Supervisor
- UC_Get_Supervisors_List

Módulo Administration

- UC_Create_UserApp
- UC_Delete_UserApp
- UC_Update_UserApp
- UC_Get_UserApp_Details
- UC_Search_UserApp
- UC_Get_List_UserApp
- UC_Send_Email

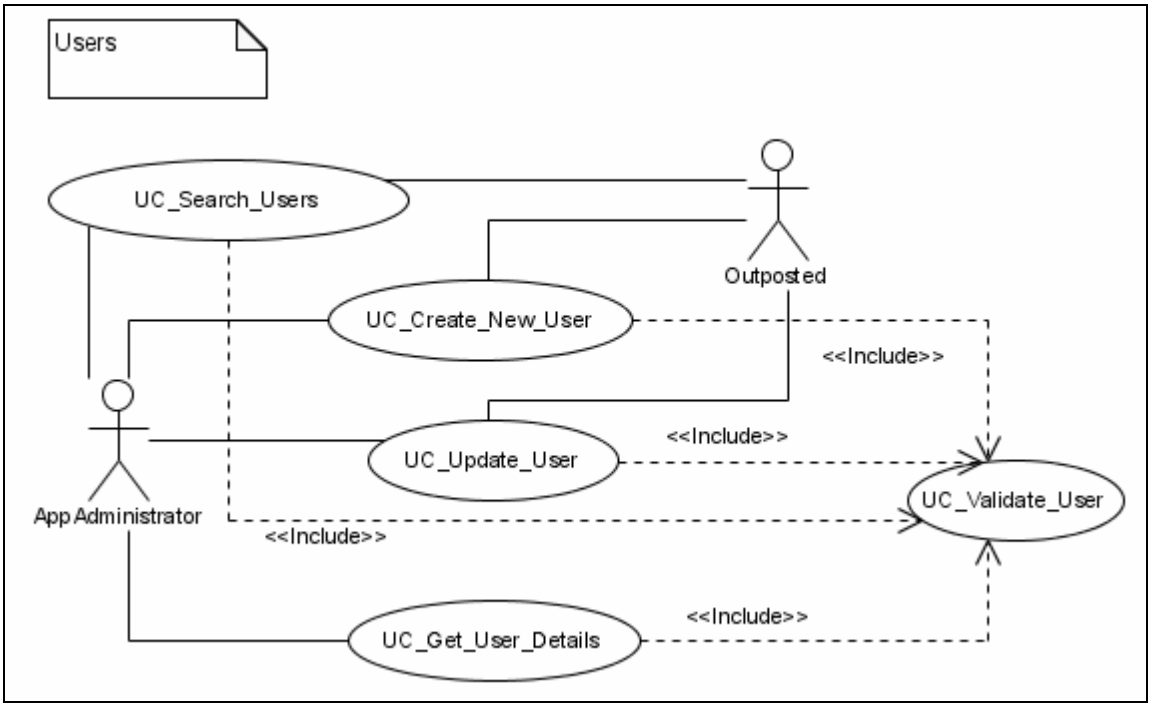
Módulo Security

- UC_Validate_User
- UC_LogIn
- UC_LogOut

7.2.3 Detalles de casos de uso. El detalle de los casos de uso se muestra por módulo o funcionalidad.

7.2.3.1 Módulo Users. Se refiere a la creación de usuarios que participan en capacitaciones, entrenamientos, eventos ofrecidos en el CIAT o en las oficinas outposted.

Figura 1. Diagrama de caso de uso de Users



- **UC_Create_New_User:** Proceso que le permite al administrador de la aplicación y al usuario outposted la creación de un nuevo usuario.

- **UC_Update_User:** Proceso que le permite al administrador de la aplicación y al usuario outposted modificar los datos de un usuario.

- **UC_Get_User_Details:** Proceso que le permite al administrador de la aplicación consultar los datos asociados a un usuario existente en la base de datos.

- **UC_Search_User:** Proceso que le permite al administrador de la aplicación y al usuario outposted la búsqueda de un usuario por nombre, apellido, o combinación de ambos o por el código que tiene asignado.

A continuación se detalla cada caso de uso:

Tabla 2. Detalle de caso de uso Create User

Nombre	**UC_Create_New_User**
Actores	-AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos información sobre los países, ciudades e instituciones y tipo de usuarios.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear un nuevo usuario. 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de un usuario. (Ver Anexo A) 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta 3. Ingresa datos o tipos de datos errados en los campos.	Muestra en pantalla un mensaje indicando los datos faltantes. Muestra en pantalla un mensaje indicando el error.
Post – condiciones	Queda almacenado en el sistema un nuevo usuario.

Tabla 3. Detalle de caso de uso Update User

Nombre	**UC_Update_User**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos usuarios creados. - Debe haber realizado una búsqueda de usuarios por nombre y/o apellido o por código.
Detalle del CU	
Actor	**Sistema**
1. Selecciona del listado de usuarios, resultado de la búsqueda, la opción para modificar un usuario. 3. Modifica los datos y selecciona la opción aceptar.	2. Muestra en pantalla los datos del usuario seleccionado. 4. Guarda las modificaciones realizadas.
Caminos alternos	
Actor	**Sistema**
3. No ingresa los datos completos 3. Ingresa datos o tipos de datos incorrectos	Muestra un mensaje en pantalla indicando los campos faltantes. Muestra en pantalla un mensaje indicando el error.
Post – condiciones	- Quedan almacenadas en el sistema las modificaciones realizadas sobre el usuario seleccionado.

Tabla 4. Detalle de caso de uso Get User Details

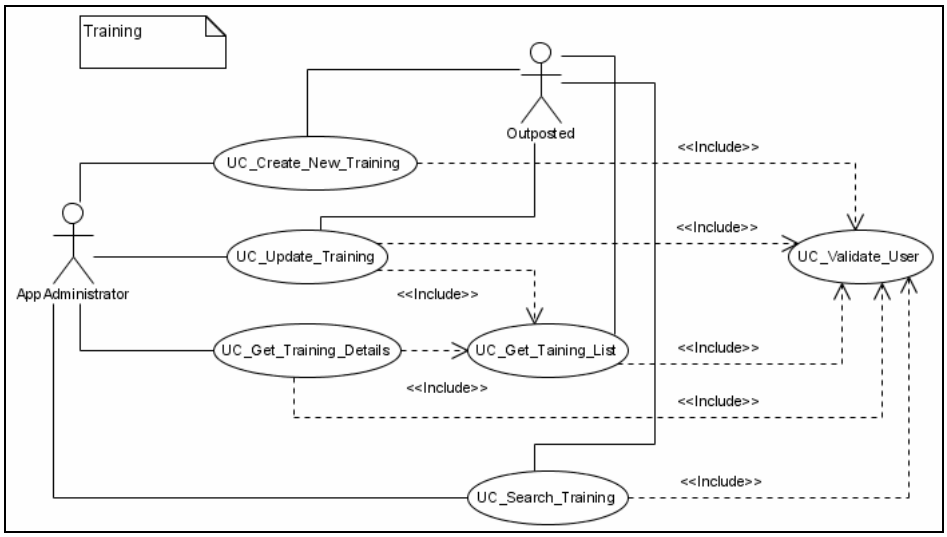
Nombre	**UC_Get_User_Details**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos usuarios creados. - Debe haber realizado una búsqueda de usuarios por nombre y/o apellido o por código.
Detalle del CU	
Actor	**Sistema**
1. Selecciona del listado de usuarios, resultado de la búsqueda, al que desea consultar los datos.	2. Muestra en pantalla los datos del usuario seleccionado.
Caminos alternos	
Actor	**Sistema**
Post – condiciones	No existen

Tabla 5. Detalle de caso de uso Search User

Nombre	**UC_Search_User**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos usuarios creados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción Users. 3. Ingresa los datos según la búsqueda que vaya a realizar y selecciona la opción buscar.	2. Muestra en pantalla los campos para realizar la búsqueda por nombre, apellido, combinación de estos y código. 4. Muestra en pantalla el listado de los usuarios que coinciden con el/los criterios de búsqueda seleccionados, con las opciones para actualizar o consultar sus detalles.
Caminos alternos	
Actor	**Sistema**
3. Digita un término de búsqueda que no coincide con ninguno de los existentes en la base de datos. 3. Ingresa un término para la búsqueda	Muestra un mensaje en pantalla indicando que para la búsqueda realizada no hubo coincidencias. Si hay resultados que coincidan con la búsqueda, y el usuario es AppAdministrator, tendrá la posibilidad de consultar y modificar cualquiera de los usuarios; si el usuario es outposted, podrá consultar los detalles de todos los usuarios resultado de la búsqueda, pero solo podrá modificar aquellos que él creó.
Post – condiciones	No existen

7.2.3.2 Módulo Training. Se refiere a los entrenamientos recibidos por los usuarios registrados en la base de datos**.**

Figura 2. Diagrama de caso de uso de Training.



- **UC_Create_New_Training:** Proceso que le permite al administrador de la aplicación y al usuario outposted la creación de un nuevo entrenamiento asociado a un usuario.

- **UC_Update_Training:** Proceso que le permite al administrador de la aplicación y al usaurio outposted modificar los datos de un entrenamiento asociado a un usuario.

- **UC_Get_Training_Details:** Proceso que le permite al administrador de la aplicación y al outposted obtener la información de los datos de un entrenamiento para un usuario determinado.

- **UC_Search_Training:** Proceso que le permite al administrador de la aplicación y al usuario outposted la búsqueda, mediante el código de usuario, de los entrenamientos realizados por un usuario.

A continuación se muestra el detalle de cada uno de los casos de uso mencionados.

Tabla 6. Detalle de caso de uso Create Training

Nombre	**UC_Create_New_Training**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos usuarios creados, países, instituciones, disciplinas.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear un nuevo entrenamiento asociado a un usuario. 3. Ingresa los datos solicitados y selecciona la opción aceptar.	2. Muestra en pantalla los campos a necesarios para la creación de un entrenamiento (Ver Anexo A). 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta 3. Ingresa datos o tipos de datos errados en los campos.	Muestra en pantalla un mensaje indicando los datos faltantes. Muestra en pantalla un mensaje indicando el error.
Post – condiciones	- Queda almacenado en el sistema un nuevo entrenamiento asociado a un usuario.

Tabla 7. Detalle de caso de uso Update Training

Nombre	**UC_Update_Training**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir almacenados en la base de datos entrenamientos asociados a usuarios. - Debe haberse realizado una búsqueda de los entrenamientos de un usuario por su código.
Detalle del CU	
Actor	**Sistema**
1. Del listado de entrenamientos realizados por un usuario, resultado de la búsqueda, selecciona el que desea modificar. 3. Modifica los datos necesarios y selecciona la opción actualizar.	2. Muestra en pantalla los datos del entrenamiento seleccionado para un usuario determinado. 4. Guarda los cambios realizados.
Caminos alternos	
Actor	**Sistema**
3. No ingresa los datos completos 3. Ingresa datos o tipos de datos incorrectos	Muestra un mensaje en pantalla indicando los campos faltantes. Muestra en pantalla un mensaje indicando el error.
Post – condiciones	- Queda almacenado en el sistema las modificaciones realizadas sobre el entrenamiento seleccionado.

Tabla 8. Detalle de caso de uso Get Training Details

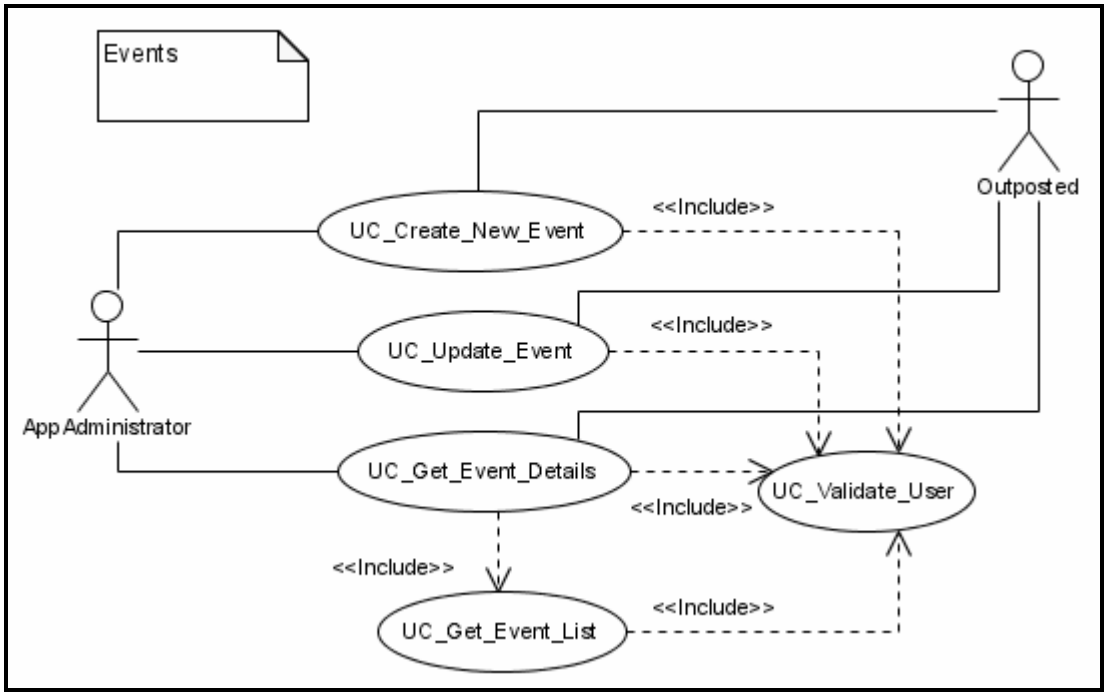
Nombre	**UC_Get_Training_Details**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir almacenados en la base de datos, la información sobre los entrenamientos recibidos por los usuarios. - Debe haberse realizado una búsqueda de los entrenamientos de un usuario por su código.
Detalle del CU	
Actor	**Sistema**
1. Del listado de entrenamientos realizados por un usuario mostrado como resultado de la búsqueda, selecciona la opción ver detalles.	2. Muestra en pantalla los datos del entrenamiento seleccionado
Caminos alternos	
Actor	**Sistema**
Post – condiciones	No existen

Tabla 9. Detalle de caso de uso Search Training

Nombre	**UC_Search_Training**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos usuarios y entrenamientos creados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción Training. 3. Ingresa el código y selecciona la opción buscar.	2. Muestra en pantalla los campos para realizar la búsqueda por código de usuario. 4. Muestra en pantalla el listado de los entrenamientos realizados por el usuario que coincide con el criterio de búsqueda seleccionado, con las opciones para actualizar o consultar sus detalles.
Caminos alternos	
Actor	**Sistema**
3. Digita código que no coincide con ninguno de los existentes en la base de datos. 3. Ingresa un término para la búsqueda	Muestra un mensaje en pantalla indicando que para la búsqueda realizada no hubo coincidencias. Si hay resultados que coincidan con la búsqueda, y el usuario es AppAdministrator, tendrá la posibilidad de consultar y modificar cualquiera de los entrenamietos listados; si el usuario es outposted, podrá consultar los detalles de todos los entrenamientos resultado de la búsqueda, pero solo podrá modificar aquellos que él creó.
Post – condiciones	No existen

7.2.3.3 Módulo Events

Figura 3. Diagrama de caso de uso de Events



- **UC_Create_New_Event:** Proceso que le permite al administrador de la aplicación la creación de un nuevo país en la base de datos.

- **UC_Update_Event:** Proceso que le permite al administrador de la aplicación modificar los datos de los países existentes en la base de datos.

- **UC_Get_Event_Details:** Proceso que le permite al administrador de la aplicación consultar los datos actuales de un país almacenado en la base de datos.

- **UC_Get_Event_List:** Proceso que le permite al administrador y al usuario outposted consultar el listado de países que se encuentran almacenados en la base de datos.

A continuación se detalla cada uno de los casos de uso de este módulo.

Tabla 10. Detalle de caso de Create New Event

Nombre	**UC_Create_New_Event**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir almacenados en la base de datos la información de países y coordinadores.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear un nuevo evento 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de un nuevo país (Ver Anexo A). 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta 3. Ingresa datos o tipos de datos errados en los campos.	Muestra en pantalla un mensaje indicando los datos faltantes. Muestra en pantalla un mensaje indicando el error.
Post – condiciones	- Queda almacenado en el sistema un nuevo país.

Tabla 11. Detalle de caso de Update Event

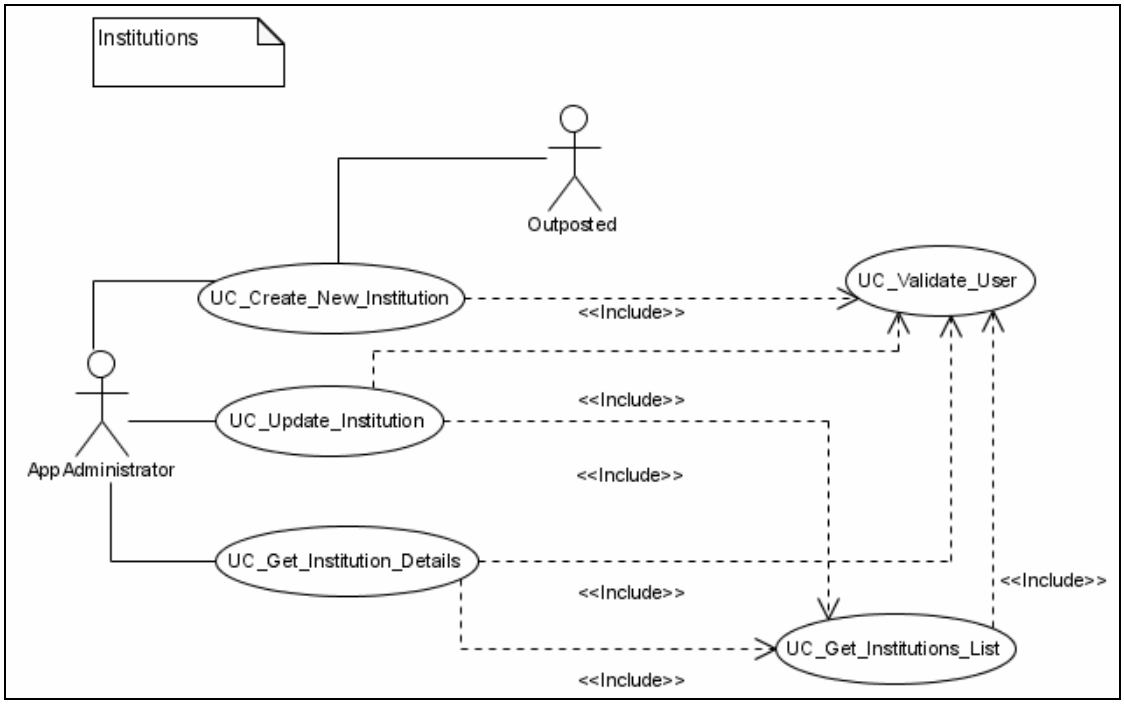
Nombre	**UC_Update_Event**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos eventos almacenados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción eventos	2. Muestra en pantalla el listado de eventos existentes en la base de datos.
3. Selecciona del listado de eventos la opción para modificar uno.	4. Muestra en pantalla los datos del evento seleccionado.
5. Modifica los datos y selecciona la opción aceptar.	6. Guarda las modificaciones realizadas.
Caminos alternos	
Actor	**Sistema**
5. No ingresa los datos completos	Muestra un mensaje en pantalla indicando los campos faltantes.
5. Ingresa datos o tipos de datos incorrectos	Muestra en pantalla un mensaje indicando el error.
Post – condiciones	- Quedan almacenadas en el sistema las modificaciones realizadas sobre el evento seleccionado.

Tabla 12. Detalle de caso de Get Event Details

Nombre	**UC_Get_Event_Details**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos eventos creados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción eventos 3. Selecciona del listado de eventos la opción ver detalles.	2. Muestra en pantalla el listado de eventos existentes en la base de datos. 4. Muestra en pantalla los datos del evento seleccionado.
Caminos alternos	
Actor	**Sistema**
Post – condiciones	No existen

7.2.3.4 Módulo Institutions

Figura 4. Diagrama de caso de uso de Institutions



- **UC_Create_New_Institution:** Proceso que le permite al administrador de la aplicación y al usuario outposted la adición de una nueva institución que realiza capacitaciones, a la base de datos.

- **UC_Update_Institution:** Proceso que le permite al administrador de la aplicación y al usuario outposted la actualización de los datos de una institución que se encuentra almacenada en la base de datos.

- **UC_Get_Institution_Details:** Proceso que le permite al administrador de la aplicación y al usuario outposted consultar la información correspondiente a una institución que se encuentra almacenada en la base de datos.
- **UC_Search_Institutions:** Proceso que le permite al administrador de la aplicación y al usuario outposted la búsqueda de una institución por su nombre.

A continuación se detalla cada uno de los casos de uso relacionados:

Tabla 13. Detalle de caso de uso Create New Institution

Nombre	**UC_Create_New_Institution**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Debe existir en la base de datos información sobre países y tipos de instituciones.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear una nueva institución. 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de una nueva institución. 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta 3. Ingresa datos o tipos de datos errados en los campos.	Muestra en pantalla un mensaje indicando los datos faltantes. Muestra en pantalla un mensaje indicando el error.
Post – condiciones	- Queda almacenado en el sistema una nueva institución

Tabla 14. Detalle de caso de uso Update Institution

Nombre	**UC_Update_Institution**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos instituciones creadas. - Se debe haber realizado una búsqueda de instituciones por nombre.
Detalle del CU	
Actor	**Sistema**
1. Del listado de instituciones mostradas resultado de la búsqueda selecciona la opción actualizar. 3. Modifica los datos deseados y selecciona la opción actualizar.	2. Muestra en pantalla la información de la institución seleccionada. 4. Guarda las modificaciones realizadas.
Caminos alternos	
Actor	**Sistema**
3. No ingresa los datos completos 3. Ingresa datos o tipos de datos incorrectos	Muestra un mensaje en pantalla indicando los campos faltantes. Muestra en pantalla un mensaje indicando el error.
Post – condiciones	- Queda almacenado en el sistema las modificaciones realizadas sobre la institución seleccionada.

Tabla 15. Detalle de caso de uso Get Institution Details

Nombre	**UC_Get_Institution_Details**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos instituciones almacenadas. - Se debe haber realizado una búsqueda de instituciones por nombre.

Detalle del CU	
Actor	**Sistema**
1. Del listado de instituciones mostradas como resultado de la búsqueda selecciona la opción ver detalles.	2. Muestra en pantalla los detalles de la institución seleccionada.
3. Selecciona la institución a consultar	4. Muestra en pantalla los datos actuales de la institución seleccionada.

Caminos alternos	
Actor	**Sistema**

Post – condiciones	No existen

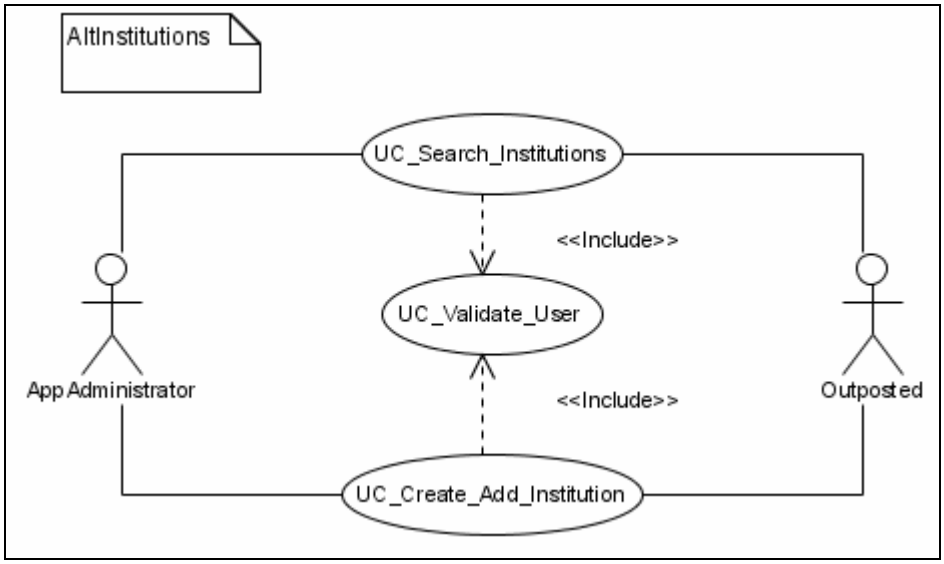
Tabla 16. Detalle de caso de uso Search Institution

<table>
<tr><td>Nombre</td><td colspan="2">UC_Search_Institution</td></tr>
<tr><td>Actores</td><td colspan="2">- AppAdministrator
- Outposted</td></tr>
<tr><td>Precondiciones</td><td colspan="2">- El usuario debe estar registrado en el sistema.
- Deben existir en la base de instituciones almacenadas.</td></tr>
<tr><td colspan="3">Detalle del CU</td></tr>
<tr><td>Actor</td><td colspan="2">Sistema</td></tr>
<tr><td>1. Selecciona la opción Institutions.

3. Ingresa el nombre y selecciona la opción buscar.</td><td colspan="2">2. Muestra en pantalla los campos para realizar la búsqueda por nombre de la institución y las opciones de búsqueda, exacta, contiene el término.
4. Muestra en pantalla el listado de las instituciones que coinciden con el criterio de búsqueda seleccionado, y las opciones para actualizar o consultar sus detalles.</td></tr>
<tr><td colspan="3">Caminos alternos</td></tr>
<tr><td>Actor</td><td colspan="2">Sistema</td></tr>
<tr><td>3. Digita un nombre que no coincide con ninguno de los existentes en la base de datos.
3. Ingresa un término para la búsqueda</td><td colspan="2">Muestra un mensaje en pantalla indicando que para la búsqueda realizada no hubo coincidencias.
Si hay resultados que coincidan con la búsqueda, y el usuario es AppAdministrator, tendrá la posibilidad de consultar y modificar cualquiera de las instituciones listados; si el usuario es outposted, podrá consultar los detalles de todas las instituciones resultado de la búsqueda, pero solo podrá modificar aquellas que él creó.</td></tr>
<tr><td>Post – condiciones</td><td colspan="2">No existen</td></tr>
</table>

7.2.3.5 Módulo AltInstitutions

Figura 5. Diagrama de caso de uso de AltInstitutions



La base de datos se diseñó de tal manera que a un entrenamiento solo se le puede asociar una Institución que se encuentre creada en la tabla ALTINSTITUTIONS, por lo tanto después de crearla según el caso de uso UC_Create_New_Institution, debe guardarse también en esta tabla.

UC_Add_New_Institution: Proceso que le permite al administrador de la aplicación adicionar una nueva institución para ser asociada al entrenamiento de un usuario.

UC_Search_Institutions: Proceso que le permite al administrador de la aplicación realizar la búsqueda de una institución que ya ha sido asociada a entrenamiento de los usuarios.

Tabla 17. Detalle de caso de uso Create New Institution

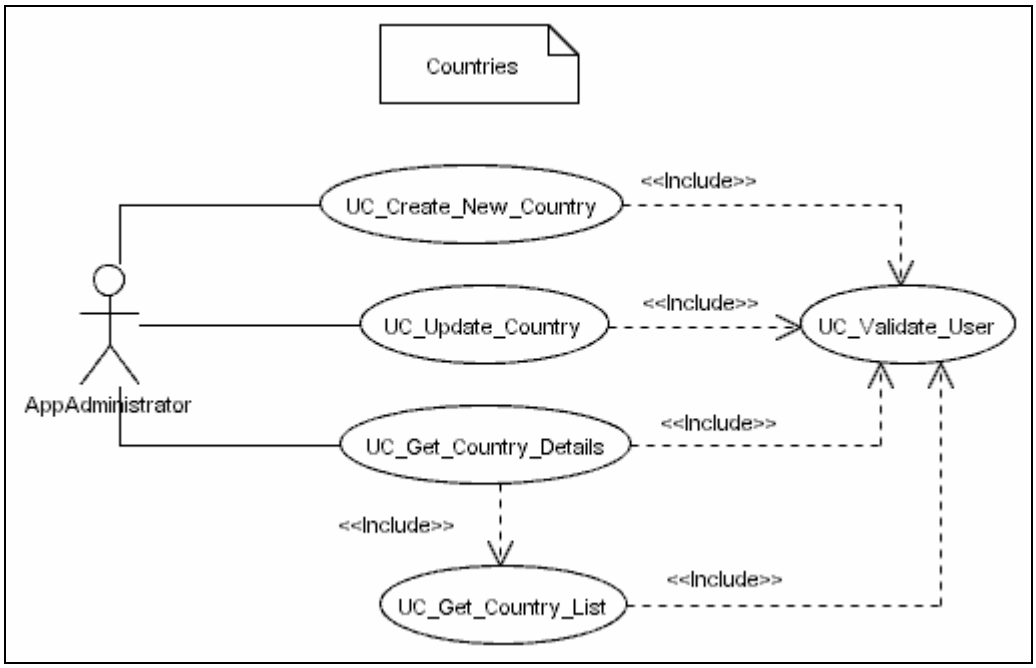
Nombre	**UC_Create_New_Institution**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Debe existir almacenada en la base de datos información sobre países.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para adicionar una nueva institución. 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de una nueva institución (Ver Anexo A). 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta 3. Ingresa datos o tipos de datos errados en los campos.	Muestra en pantalla un mensaje indicando los datos faltantes. Muestra en pantalla un mensaje indicando el error.
Post – condiciones	- Queda almacenado en el sistema una nueva institución

Tabla 18. Detalle de caso de uso Search Institution

Nombre	**UC_Search_Institution**
Actores	- AppAdministrator - Outposted
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de instituciones almacenadas.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción Institutions. 3. Ingresa el nombre y selecciona la opción buscar.	2. Muestra en pantalla los campos para realizar la búsqueda por nombre de la institución y las opciones de búsqueda, exacta, contiene el término. 4. Muestra en pantalla el listado de las instituciones que coinciden con el criterio de búsqueda seleccionado, y las opciones para actualizar o consultar sus detalles.
Caminos alternos	
Actor	**Sistema**
3. Digita un nombre que no coincide con ninguno de los existentes en la base de datos. 3. Ingresa un término para la búsqueda	Muestra un mensaje en pantalla indicando que para la búsqueda realizada no hubo coincidencias. Si hay resultados que coincidan con la búsqueda, y el usuario es AppAdministrator, tendrá la posibilidad de consultar y modificar cualquiera de las instituciones listados; si el usuario es outposted, podrá consultar los detalles de todas las instituciones resultado de la búsqueda, pero solo podrá modificar aquellas que él creó.
Post – condiciones	No existen

7.2.3.6 Módulo Countries

Figura 6. Diagrama de caso de uso de Countries



- **UC_Create_New_Countries:** Proceso que le permite al administrador de la aplicación la creación de un nuevo país en la base de datos..

- **UC_Update_Countries:** Proceso que le permite al administrador de la aplicación modificar los datos de los países existentes en la base de datos. (R-PA2)

- **UC_Get_Countries_Details:** Proceso que le permite al administrador de la aplicación consultar los datos actuales de un país almacenado en la base de datos. (R-PA3)

- **UC_Search_Country:** Proceso que le permite al administrador de la aplicación la búsqueda de países por nombre.

- **UC_Get_Country_List:** Proceso que le permite al administrador y al usuario outposted consultar el listado de países que se encuentran almacenados en la base de datos.

A continuación se detallan cada uno de los casos de uso mencionados.

Tabla 19. Detalle de caso de uso Create New Country

Nombre	**UC_Create_New_Country**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir almacenados en la base de datos la información de los continentes.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear un nuevo país 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de un nuevo país (Ver Anexo A). 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta	Muestra en pantalla un mensaje indicando los datos faltantes.
Post – condiciones	- Queda almacenado en el sistema un nuevo país.

Tabla 20. Detalle de caso de uso Update Country

Nombre	**UC_Update_Country**
Actores	- AppAdministrator -
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos países creados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción países 3. Selecciona del listado de países la opción para modificar uno. 5. Modifica los datos y selecciona la opción aceptar.	2. Muestra en pantalla el listado de países existentes en la base de datos. 4. Muestra en pantalla los datos del país seleccionado. 6. Guarda las modificaciones realizadas.
Caminos alternos	
Actor	**Sistema**
Post – condiciones	- Queda almacenado en el sistema las modificaciones realizadas en el país seleccionado.

Tabla 21. Detalle de caso de uso Get Country Details

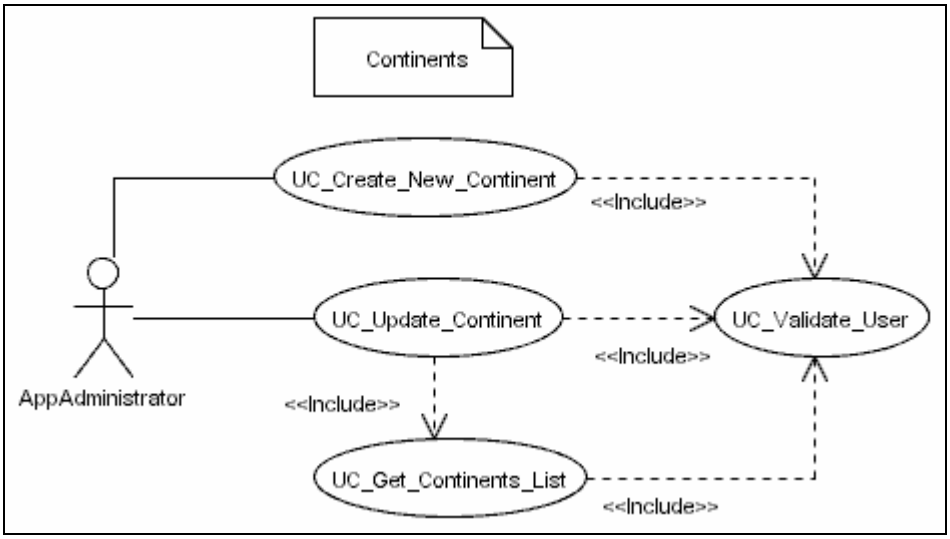
Nombre	**UC_Get_Country_Details**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos países almacenados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción países 3. Selecciona del listado de países la opción ver detalles.	2. Muestra en pantalla el listado de países existentes en la base de datos. 4. Muestra en pantalla los datos del país seleccionado.
Caminos alternos	
Actor	**Sistema**
Post – condiciones	No existen.

Tabla 22. Detalle de caso de uso Search Country

Nombre	**UC_Search_Country**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos países almacenados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción buscar país. 3. Ingresa el nombre y selecciona la opción buscar.	2. Muestra en pantalla los campos para realizar la búsqueda por nombre del país y las opciones de búsqueda, exacta, contiene el término. 4. Muestra en pantalla el listado de los países que coinciden con el criterio de búsqueda seleccionado, y la opción para actualizar.
Caminos alternos	
Actor	**Sistema**
3. Digita un nombre que no coincide con ninguno de los existentes en la base de datos.	Muestra un mensaje en pantalla indicando que para la búsqueda realizada no hubo coincidencias.
Post – condiciones	No existen

7.2.3.7 Módulo Continents

Figura 7. Diagrama de caso de uso de Continents



- **UC_Create_New_Continents:** Proceso mediante el administrador de la aplicación puede crear en la base de datos de un nuevo continente.

- **UC_Update_Continents:** Proceso mediante el cual el administrador de la aplicación puede actualizar los datos del continente seleccionado.

- **UC_Get_Continents_List:** Proceso que le permite al administrador consultar el listado de los continentes que se encuentran almacenados en la base de datos.

A continuación se detallan los casos de uso mencionados.

Tabla 23. Detalle de caso de uso Create New Continent

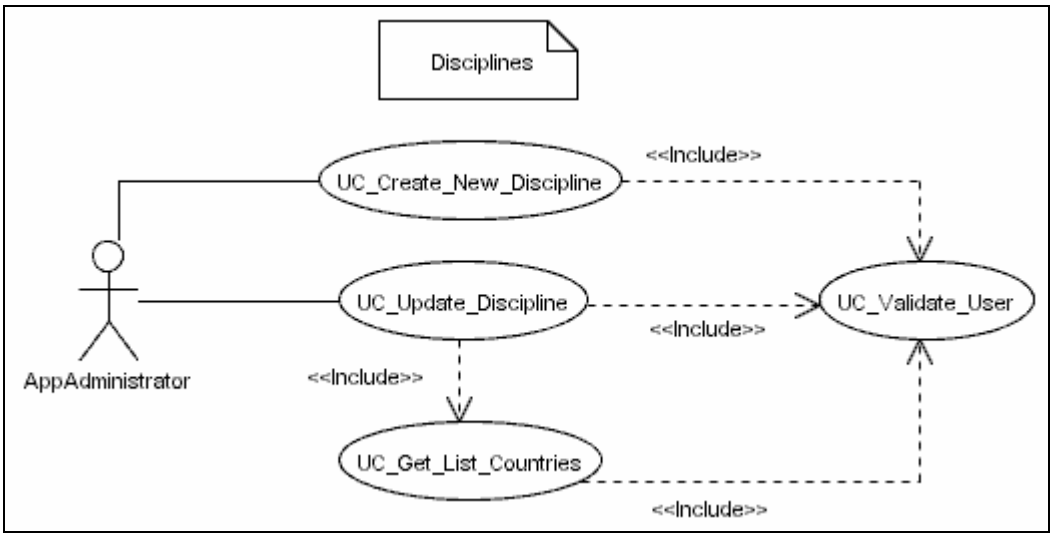
Nombre	**UC_Create_New_Continent**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear un nuevo continente. 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de un nuevo continente (Ver Anexo A). 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta	Muestra en pantalla un mensaje indicando los datos faltantes.
Post – condiciones	- Queda almacenado en el sistema un nuevo continente.

Tabla 24. Detalle de caso de uso Update Continent

Nombre	**UC_Update_Continent**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos continentes creados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para modificar un continente.	2. Muestra en pantalla el listado de los continentes existentes en la base de datos.
3. Selecciona el continente a modificar.	4. Muestra en pantalla los campos que se pueden modificar del continente.
5. Modifica los datos y selecciona la opción aceptar.	6. Guarda las modificaciones realizadas.
Caminos alternos	
Actor	**Sistema**
Post – condiciones	- Queda almacenado en el sistema las modificaciones realizadas en el continente seleccionado.

7.2.3.8 Módulo Disciplines

Figura 8. Diagrama de caso de uso de Disciplines



- **UC_Create_New_Disciplines:** Proceso que le permite al administrador de la aplicación adicionar nuevas disciplinas en la base de datos.

- **UC_Update_Disciplines:** Proceso que le permite al administrador del sistema modificar los datos de las disciplinas que se encuentran almacenadas en la base de datos.

- **UC_Search_Disciplines:** Proceso que le permite al administrador de la aplicación la búsqueda de disciplinas por nombre.

- **UC_Get_Disciplines_List:** Proceso que le permite al adminsitrador de la aplicación el listado de las disciplinas que se encuentran en la base de datos.

A continuación se detalla cada uno de los casos de uso mencionados.

Tabla 25. Detalle de caso de uso Create New Discipline

Nombre	**UC_Create_New_Discipline**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear una nueva disciplina. 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de una nueva disciplina (Ver Anexo A). 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta	Muestra en pantalla un mensaje indicando los datos faltantes.
Post – condiciones	- Queda almacenado en el sistema una nueva disciplina.

Tabla 26. Detalle de caso de uso Update Discipline

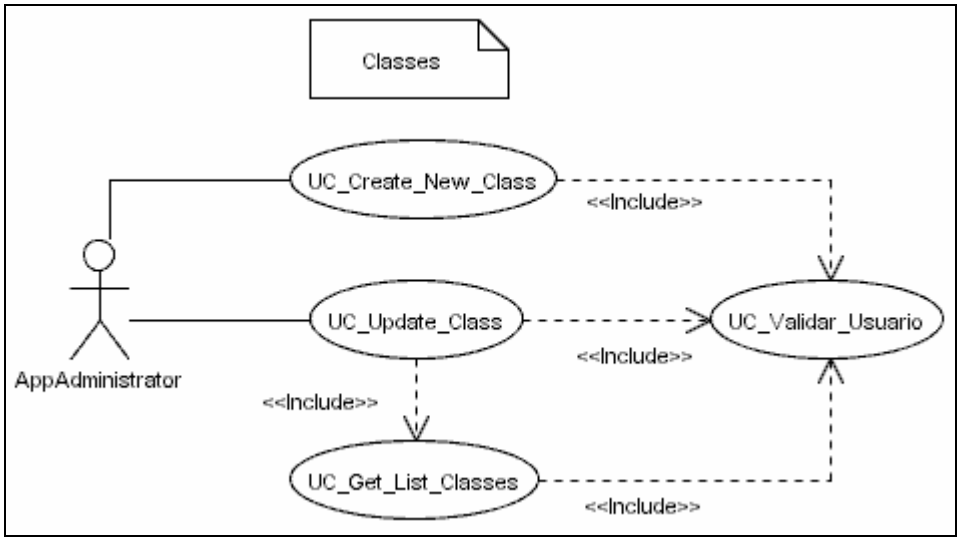
Nombre	**UC_Update_Discipline**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos disciplinas creadas.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para modificar una disciplina.	2. Muestra en pantalla el listado de las disciplinas existentes en la base de datos.
3. Selecciona la disciplina a modificar.	4. Muestra en pantalla los campos que se pueden modificar de la disciplina.
5. Modifica los datos y selecciona la opción aceptar.	6. Guarda las modificaciones realizadas.
Caminos alternos	
Actor	**Sistema**
5. Modifica el dato	Si la disciplina está asociada a algún entrenamiento, no permitirá realizar el cambio.
Post – condiciones	- Queda almacenado en el sistema las modificaciones realizadas en la disciplina seleccionada.

Tabla 27. Detalle de caso de uso Search Discipline

Nombre	**UC_Search_Discipline**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos disciplinas almacenados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción buscar disciplina. 3. Ingresa el nombre y selecciona la opción buscar.	2. Muestra en pantalla los campos para realizar la búsqueda por nombre de la disciplina y las opciones de búsqueda, exacta, contiene el término. 4. Muestra en pantalla el listado de las disciplinas que coinciden con el criterio de búsqueda seleccionado, y la opción para actualizar.
Caminos alternos	
Actor	**Sistema**
3. Digita un nombre que no coincide con ninguno de los existentes en la base de datos.	Muestra un mensaje en pantalla indicando que para la búsqueda realizada no hubo coincidencias.
Post – condiciones	No existen

7.2.3.9 Módulo Classes

Figura 9. Diagrama de caso de uso de Classes



- **UC_Create_New_Class:** Proceso que le permite al administrador de la aplicación la creación de nuevas clases.

- **UC_Update_Class:** Proceso que le permite al administrador de la aplicación modificar los datos de las clases que se encuentran almacenados en la base de datos.

- **UC_Search_Class:** Proceso que le permite al administrador de la aplicación la búsqueda de clases por nombre.

- **UC_Get_Class_List:** Proceso que le permite al admistrador de la aplicación consultar el listado de las clases que se encuentran almacenadas en la base de datos.

A continuación se detalla cada uno de los casos de uso del módulo.

Tabla 28. Detalle de caso de uso Create New Class

Nombre	**UC_Create_New_Class**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear una nueva clase. 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de una nueva clase (Ver Anexo A). 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta	Muestra en pantalla un mensaje indicando los datos faltantes.
Post – condiciones	- Queda almacenado en el sistema una nueva clase.

Tabla 29. Detalle de caso de uso Update Class

Nombre	**UC_Update_Class**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos clases creadas.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para modificar una clase.	2. Muestra en pantalla el listado de las clases existentes en la base de datos.
3. Selecciona la clase a modificar.	4. Muestra en pantalla los campos que se pueden modificar de la clase.
5. Modifica los datos y selecciona la opción aceptar.	6. Guarda las modificaciones realizadas.
Caminos alternos	
Actor	**Sistema**
5. Modifica el dato	Si la clase está asociada a alguna información de usuarios, no se permitirá realizar el cambio.
Post – condiciones	- Queda almacenado en el sistema las modificaciones realizadas en la clase seleccionada.

Tabla 30. Detalle de caso de uso Search Class

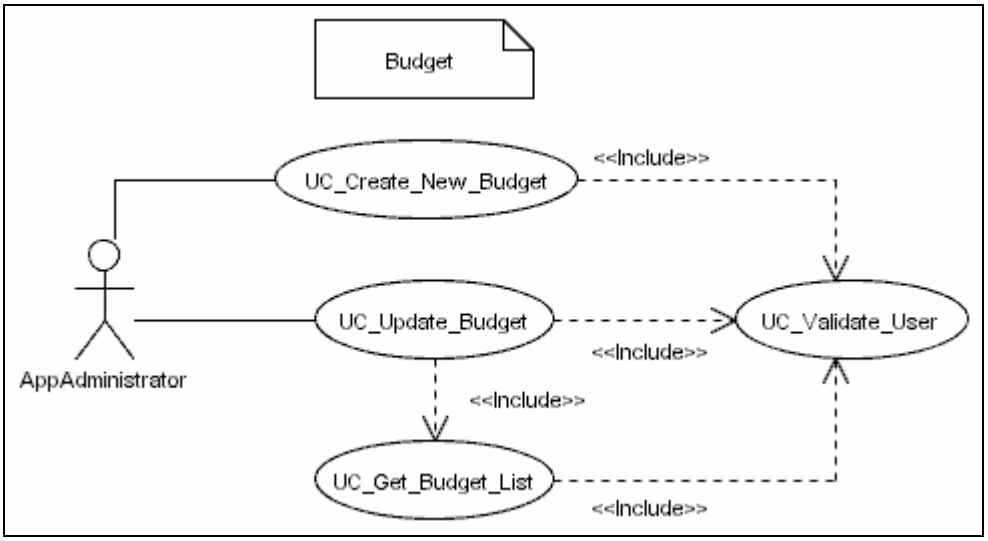
<table>
<tr><td>Nombre</td><td colspan="2">UC_Search_Class</td></tr>
<tr><td>Actores</td><td colspan="2">- AppAdministrator</td></tr>
<tr><td>Precondiciones</td><td colspan="2">- El usuario debe estar registrado en el sistema.
- Deben existir en la base de datos clases almacenados.</td></tr>
<tr><td colspan="3">Detalle del CU</td></tr>
<tr><td colspan="2">Actor</td><td>Sistema</td></tr>
<tr><td colspan="2">1. Selecciona la opción buscar clase.

3. Ingresa el nombre y selecciona la opción buscar.</td><td>2. Muestra en pantalla los campos para realizar la búsqueda por nombre de la clase y las opciones de búsqueda, exacta, contiene el término.

4. Muestra en pantalla el listado de las clases que coinciden con el criterio de búsqueda seleccionado, y la opción para actualizar.</td></tr>
<tr><td colspan="3">Caminos alternos</td></tr>
<tr><td colspan="2">Actor</td><td>Sistema</td></tr>
<tr><td colspan="2">3. Digita un nombre que no coincide con ninguno de los existentes en la base de datos.</td><td>Muestra un mensaje en pantalla indicando que para la búsqueda realizada no hubo coincidencias.</td></tr>
<tr><td>Post – condiciones</td><td colspan="2">No existen</td></tr>
</table>

7.2.3.10 Módulo Budgets

Figura 10. Diagrama de caso de uso de Budgets



- **UC_Create_New_Budgets:** Proceso que le permite al administrador de la aplicación adicionar en la base de datos nuevos presupuestos.

- **UC_Update_Budgets:** Proceso que le permite al administrador de la aplicación modificar los datos de los presupuestos que se encuentran almacenados en la base de datos.

- **UC_Get_Budget_List:** Proceso que le permite al administrador de la aplicación consultar el listado de los presupuestos que se manejan en el CIAT y que se encuentran almacenados en la base de datos.

A continuación se detallan los casos de uso correspondientes a este módulo.

Tabla 31. Detalle de caso de uso Create New Budget

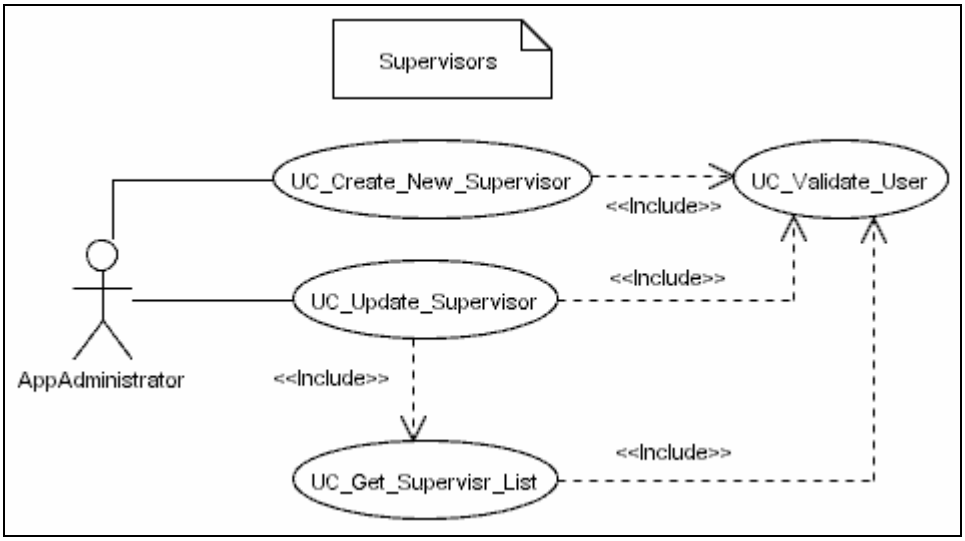
Nombre	**UC_Create_New_Budget**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear un nuev presupuesto. 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de un nuevo presupuesto (Ver Anexo A). 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta	Muestra en pantalla un mensaje indicando los datos faltantes.
Post – condiciones	- Queda almacenado en el sistema un nuevo presupuesto.

Tabla 32. Detalle de caso de uso Update Budget

Nombre	**UC_Update_Budget**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos presupuestos creadas.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para modificar un presupuesto.	2. Muestra en pantalla el listado de los presupuestos existentes en la base de datos.
3. Selecciona el presupuesto a modificar.	4. Muestra en pantalla los campos que se pueden modificar del presupuesto.
5. Modifica los datos y selecciona la opción aceptar.	6. Guarda las modificaciones realizadas.
Caminos alternos	
Actor	**Sistema**
5. Modifica el dato	Si el presupuesto está asociado a alguna información, no permitirá realizar el cambio.
Post – condiciones	- Queda almacenado en el sistema las modificaciones realizadas en el presupuesto seleccionado.

7.2.3.11 Módulo Supervisors

Figura 11. Diagrama de caso de uso de Supervisors



- **UC_Create_New_Supervisor:** Proceso que le permite al administrador de la aplicación adicionar nuevos supervisores en la base de datos.

- **UC_Update_Supervisors:** Proceso que le permite al administrador del sistema actualizar los datos de los supervisores existentes en la base de datos. (R-S2)

- **UC_Search_Supervisor:** Proceso que le permite al administrador de la aplicación la búsqueda de supervisores por nombre.

- **UC_Get_Supervisors_List:** Proceso que le permite al adminsitrador de la aplicación consultar la lista de los supervisores asignados a los entrenamientos y que se encuentran almacenados en la base de datos.

Tabla 33. Detalle de caso de uso Create New Supervisor

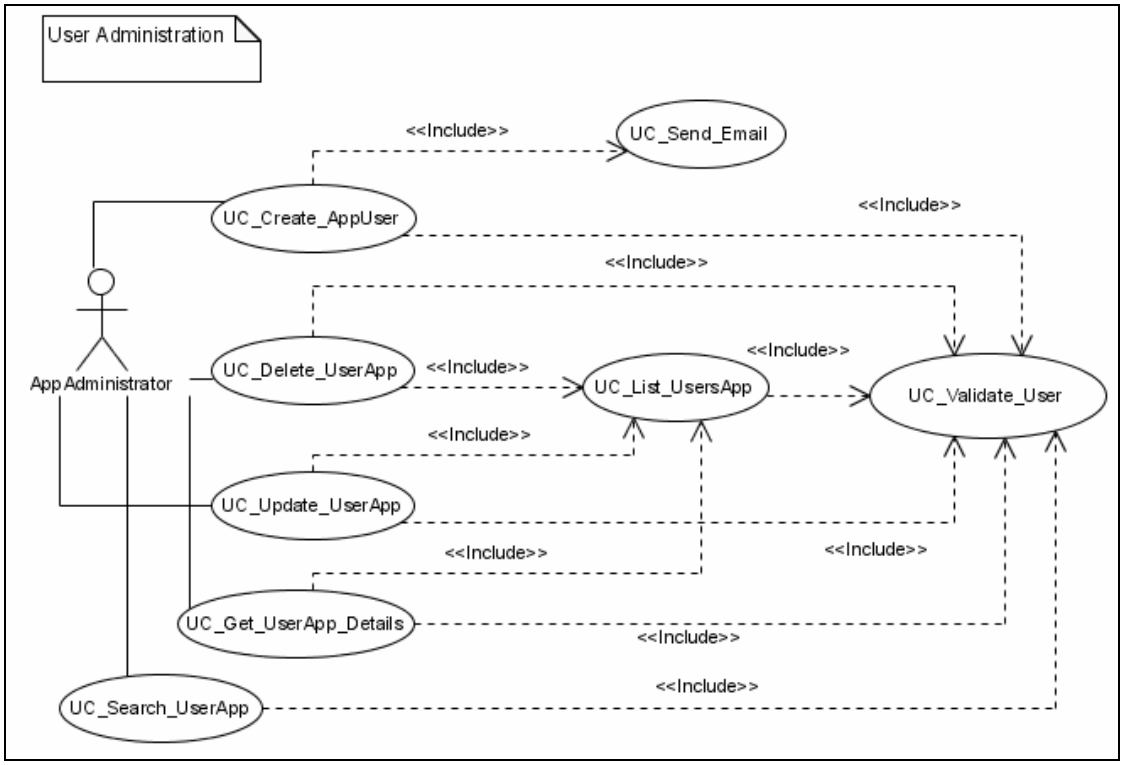
Nombre	**UC_Create_New_Supervisor**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear un nuevo supervisor. 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de un nuevo supervisor (Ver Anexo A). 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta	Muestra en pantalla un mensaje indicando los datos faltantes.
Post – condiciones	- Queda almacenado en el sistema un nuevo supervisor.

Tabla 34. Detalle de caso de uso Update Supervisor

Nombre	**UC_Update_Supervisor**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos supervisores creados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para modificar un supervisor.	2. Muestra en pantalla el listado de los supervisores existentes en la base de datos.
3. Selecciona el supervisor a modificar.	4. Muestra en pantalla los campos que se pueden modificar del supervisor, si no está asociado a algún entrenamiento.
5. Modifica los datos y selecciona la opción aceptar.	6. Guarda las modificaciones realizadas.
Caminos alternos	
Actor	**Sistema**
5. Modifica el dato	Si el supervisor está asociado a alguna información de usuarios, no se permitirá realizar el cambio.
Post – condiciones	- Queda almacenado en el sistema las modificaciones realizadas en el supervisor seleccionado.

7.2.3.12 Módulo Administration

Figura 12. Diagrama de caso de uso de Administration



- **UC_Create_UserApp:** Proceso que le permite al administrador de la aplicación la creación de un usuario de la aplicación asignándole un rol determinado.

- **UC_Delete_UserApp:** Proceso que le permite al administrador de la aplicación borrar un usuario de la aplicación.

- **UC_Send_Email:** Proceso que permite que el sistema envíe una notificación los usuarios cuando el administrador ha creado su perfil en la aplicación y asignandole password y login.

- **UC_Update_UserApp:** Proceso que le permite al usuario outposted modificar los datos de su cuenta, y al administrador de la aplicación modificar tanto sus datos como los de cualquier usuario outposted.

- **UC_Get_UserApp_Details:** Proceso que le permite al administrador de la aplicación consultar los datos personales de los usuarios de la aplicación creados.

- **UC_List_UsersApp:** Proceso que permite al administrador de la aplicación listar los usuarios de la aplicación, es decir los outposted y los administradores que se encuentran creados en la base de datos.

A continuación se detallan los casos de uso correspondientes a este módulo.

Tabla 35. Detalle de caso de Create AppUser

Nombre	**UC_Create_AppUser**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción para crear un nuevo usuario de la aplicación. 3. Ingresa la información solicitada y selecciona la opción aceptar	2. Muestra en pantalla los campos a llenar para la creación de un nuevo usuario (Ver Anexo A). 4. Guarda los datos.
Caminos alternos	
Actor	**Sistema**
3. Ingresa información incompleta 3. Ingresa datos o tipos de datos errados en los campos.	Muestra en pantalla un mensaje indicando los datos faltantes. Muestra en pantalla un mensaje indicando el error.
Post – condiciones	- Queda almacenado en el sistema un nuevo usuario de la aplicación.

Tabla 36. Detalle de caso de Update AppUser

Nombre	**UC_Update_UserApp**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos usuarios creados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción Administración	2. Muestra en pantalla el listado de los presupuestos existentes en la base de datos.
3. Selecciona el usuario a modificar.	4. Muestra en pantalla los campos que se pueden modificar del usuario.
5. Modifica los datos y selecciona la opción aceptar.	6. Guarda las modificaciones realizadas.
Caminos alternos	
Actor	**Sistema**
Post – condiciones	- Queda almacenado en el sistema las modificaciones realizadas en el usuario seleccionado.

Tabla 37. Detalle de caso de Get AppUser Details

Nombre	**UC_Get_AppUser_Details**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos usuarios almacenados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción administración 3. Selecciona del listado de usuarios la opción ver detalles.	2. Muestra en pantalla el listado de usuarios de la aplicación existentes en la base de datos. 4. Muestra en pantalla los datos del usuario seleccionado.
Caminos alternos	
Actor	**Sistema**
Post – condiciones	No existen.

Tabla 38. Detalle de caso de Delete AppUser

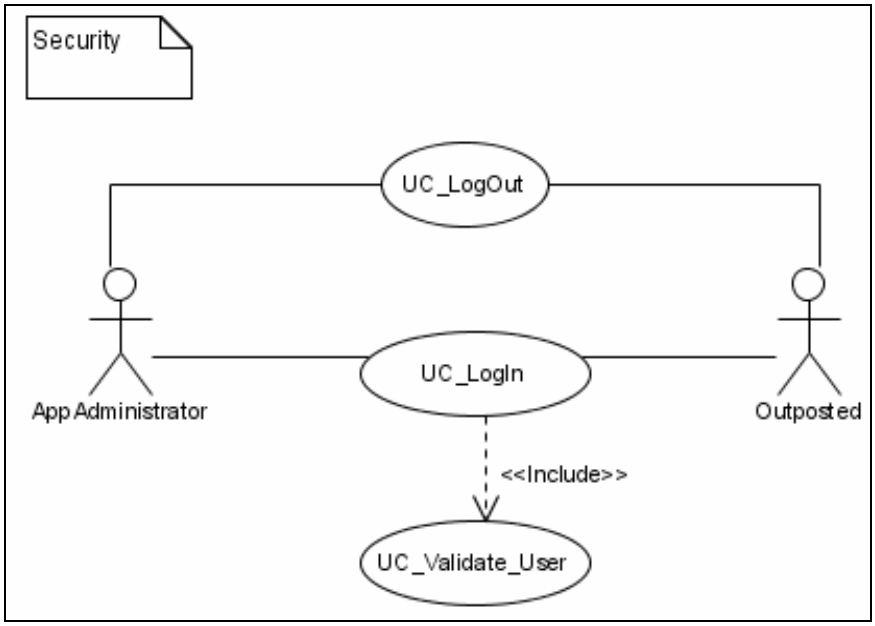
Nombre	**UC_Delete_AppUser**
Actores	- AppAdministrator
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos usuarios almacenados.
Detalle del CU	
Actor	**Sistema**
1. Selecciona la opción administración	2. Muestra en pantalla el listado de usuarios de la aplicación existentes en la base de datos.
3. Selecciona del listado de usuarios la opción ver borrar.	4. Muestra en pantalla un mensaje para confirmar la eliminación del usuario.
5. Selecciona la opción aceptar.	6. Elimina el usuario
Caminos alternos	
Actor	**Sistema**
5. Selecciona la opción no borrar.	Queda a la espera de una nueva acción.
Post – condiciones	Se elimina de la base de datos el registro del usuario.

Tabla 39. Detalle de caso de Send Email

Nombre	**UC_Send_Email**
Actores	- UC_Create_AppUser
Precondiciones	- El usuario debe estar registrado en el sistema. - Deben existir en la base de datos usuarios almacenados.
Detalle del CU	
Actor	**Sistema**
1. Envía los datos del nuevo usuario creado: nombre, password, username.	2. Toma los datos, los agrupa y los agrega al mensaje predeterminado que indica al usuario creado la activación de su cuenta y envía un mensaje de correo al usuario y una copia al administrador de la aplicación (el que creó al usuario)
Caminos alternos	
Actor	**Sistema**
Post – condiciones	Un email es enviado al nuevo usuario

17.2.3.13 Módulo Security

Figura 13. Diagrama de caso de uso de Security



- **UC_Validate_User:** Proceso que permite al sistema validar si un usuario existe en la base de datos para permitirle el acceso a la aplicación.

- **UC_LogIn**: Proceso que les permite a los usuarios ingresar password y login para que le sea permitido el ingreso al sistema.

- **UC_LogOut:** Proceso que permite al usuario salir de la aplicación.

A continuación se detallan los casos de uso correspondientes a este módulo:

Tabla 40. Detalle de caso de Uso LogIn

Nombre	**UC_Log_In**
Actores	- AppUser - Outposted
Precondiciones	- Deben existir en la base de datos usuarios almacenados.
Detalle del CU	
Actor	**Sistema**
1. El usuario digita el login y el password 3. Ingresa al sistema.	2. Toma los datos y usa el caso de uso UC_Validate_User
Caminos alternos	
Actor	**Sistema**
Post – condiciones	No Existen.

Tabla 41. Detalle de caso de Uso LogOut

Nombre	**UC_Log_Out**	
Actores	- AppUser - Outposted	
Precondiciones	- Deben existir en la base de datos usuarios almacenados.	
Detalle del CU		
Actor		**Sistema**
1. El usuario selecciona la opción Log Out del menú.		2. Finaliza la sesión y muestra la página de inicio de la aplicación.
Caminos alternos		
Actor		**Sistema**
Post – condiciones	No Existen.	

Tabla 42. Detalle de caso de Uso Validate User

Nombre	**UC_Validate_User**	
Actores	- UC_LogIn	
Precondiciones	- Deben existir en la base de datos usuarios almacenados.	
Detalle del CU		
Actor		**Sistema**
		1. Toma los datos del login y el password y los compara con la base de datos. Si coinciden permite el ingreso a la aplicación, de lo contrario muestra un mensaje en pantalla indicando el error.
Caminos alternos		
Actor		**Sistema**
Post – condiciones	No Existen.	

8. DISEÑO Y CONSTRUCCIÓN

8.1 DISEÑO DEL SOFWARE

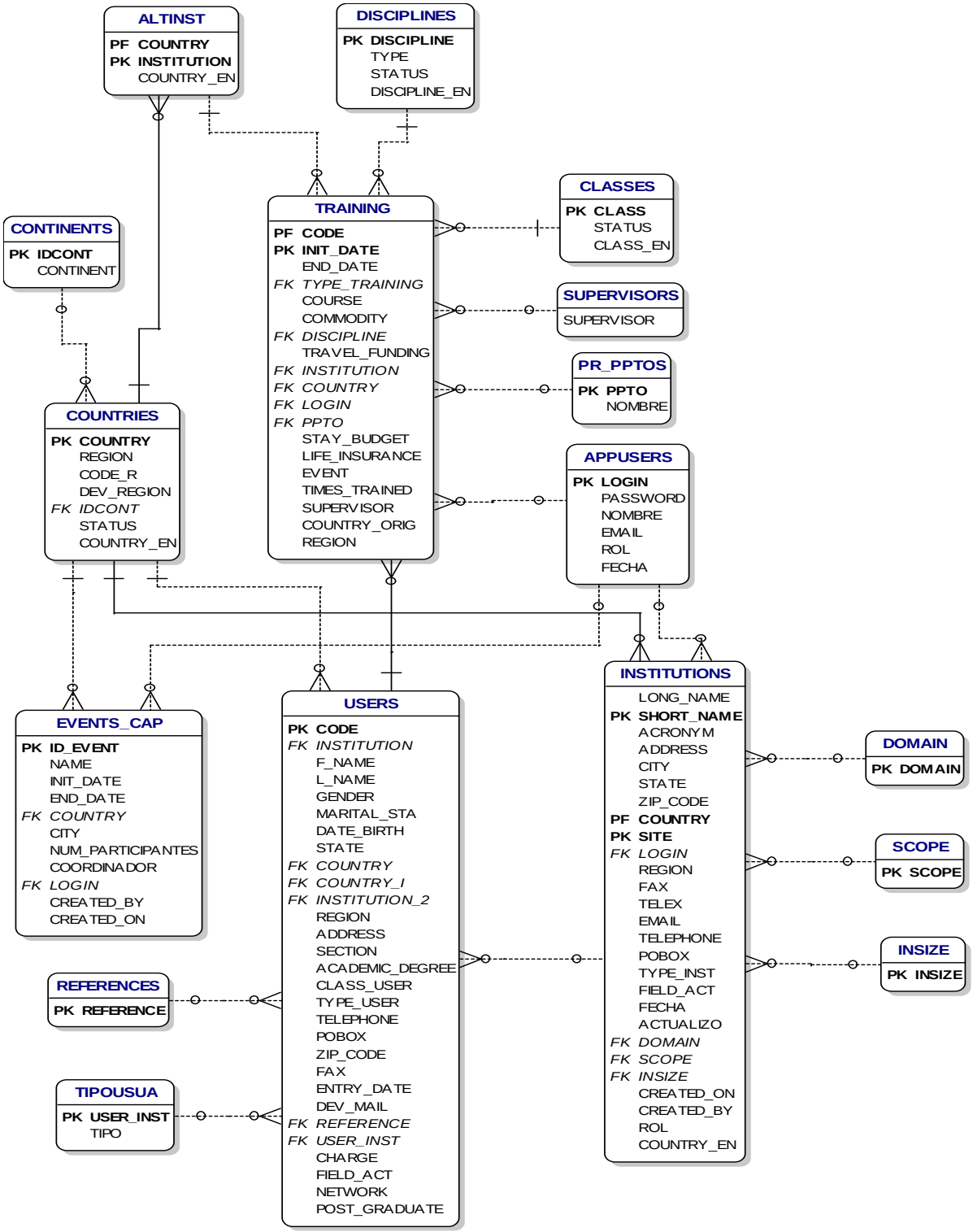
Durante esta etapa del desarrollo del proyecto se realizaron los cambios necesarios al modelo entidad relación (MER) existente. La figura ¿?? Ilustra el modelo final.

La metodología empleada en Ingeniería de Software que utiliza el Proceso Unificado de Desarrollo UP en conjunto con el lenguaje de modelado unificado, UML especifica que en esta etapa se deben elaborar diagramas de clase y de secuencia. Dada la necesidad del cliente de que se realizara la aplicación en el menor tiempo posible, el equipo de desarrollo decidió emplear DBForms, un framework o herramienta de desarrollo rápido para Java (JSP/Servlets/Taglib) para cumplir con los tiempos estimados.

DBForms se emplea para el desarrollo de aplicaciones en las cuales gran parte del desempeño de la misma se centra en la base de datos, y le permite al usuario localizar componentes y elementos de acción como botones en plantillas, que luego son ejecutadas en tiempo de ejecución.

Uno de los grandes beneficios que se desprenden del uso de DBForms es que tiene una arquitectura abierta, puede ser usado en conjunto con JSP, Struts, entre otros, además de que reduce el tiempo de desarrollo, pero dada su arquitectura no es necesaria la realización de diagramas de clases ni de secuencia, pues no maneja el concepto de clases: el framework tiene un mecanismo mediante el cual se mapean las tablas de la base de datos y mediante el uso de un controlador interno se realizan las diferentes acciones como insertar, actualizar y eliminar información de la base de datos, esta es la razón por la cual este documento carece de los diagramas mencionados.

Figura 14. Modelo Entidad Relación



8.2 CONSTRUCCIÓN DEL SOFTWARE

La Unidad de Sistemas de Información del CIAT cuenta con una plataforma definida para el desarrollo de software conformada por los siguientes elementos:

- Lenguaje de programación: Para todos los desarrollos *web* se emplea el lenguaje de programación Java, en este caso se empleó el SDK 1.4.2_09.

- Framework DBForms: para el desarrollo rápido de aplicaciones, el cual está basado en las especificaciones Java Servlets 2.3 y Java Serves Pages 1.2 de Sun Microsystems.

- Entorno de desarrollo IDE: se emplea Eclipse, pues ofrece grandes ventajas como el hecho de ser gratuito y de contar con un alto número de *plugins* disponibles que aumentan su funcionalidad. MyEclipse es uno de los plugins que extienden las capacidades de Eclipse para el desarrollo J2EE, lo cual permite desarrollar aplicaciones web que incluyen JSP, JSF, Struts, XML, Servlets, EJB.

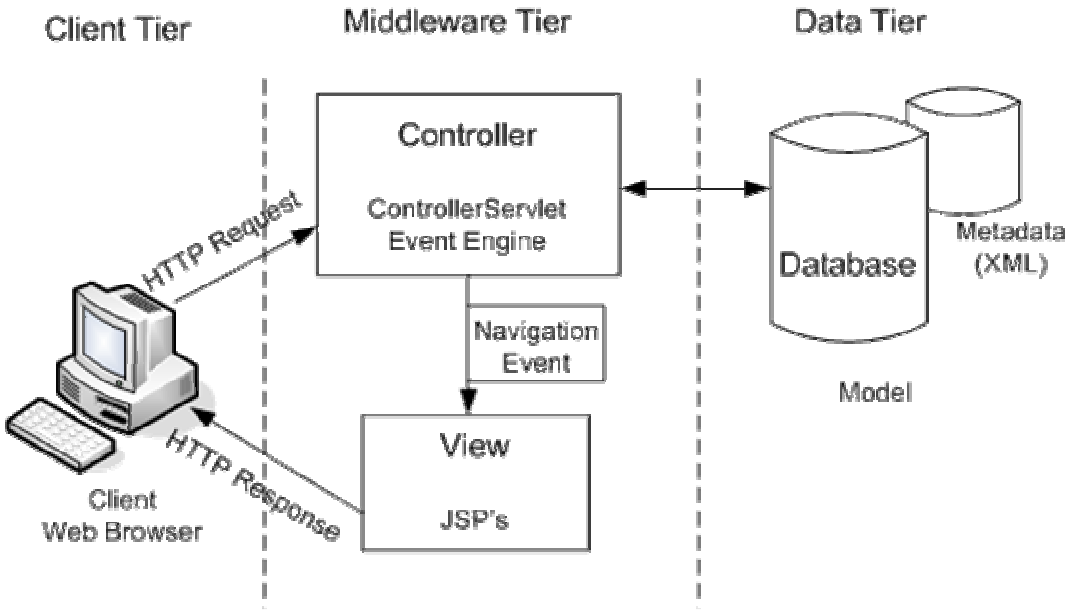
- Motor de Bases de Datos: el estándar es ORACLE, para el proyecto se empleó la versión 9i.

- Servidor de aplicaciones: se emplea Tomcat como contenedor de servlets. El equipo sobre el cual corren las aplicaciones cuenta con Apache Server.

8.3 DIAGRAMA DE LA ARQUITECTURA SOBRE LA QUE QUEDÓ FUNCIONANDO EL SISTEMA

La arquitectura del sistema es de tres capas, ya que DBForms implementa los conceptos del patrón de diseño Modelo Vista Controlador. La Figura 15 ilustra la arquitectura del sistema.

Figura 15. Diagrama de la arquitectura del sistema

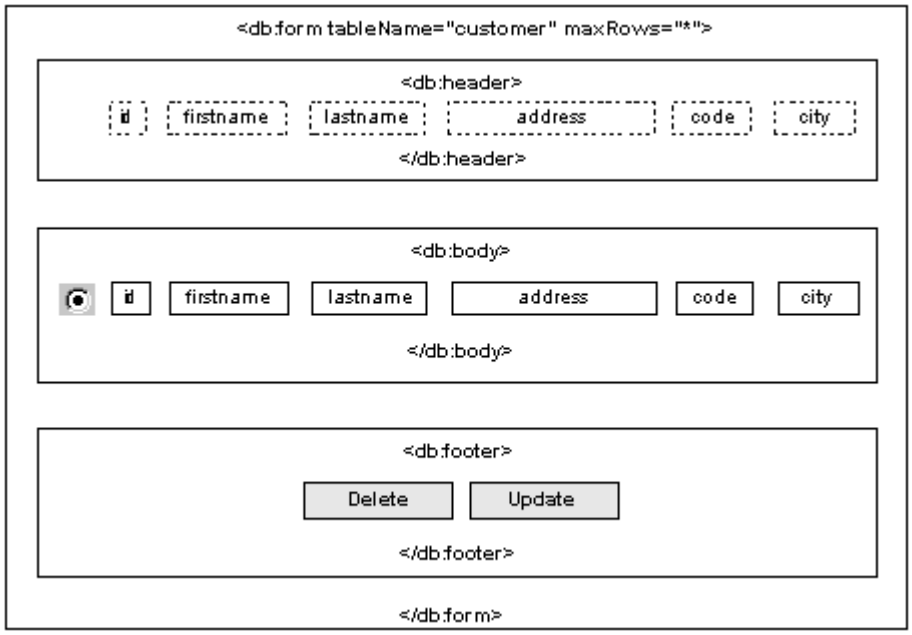


En este caso la Vista está representada por las JSP, el Modelo por las tablas de la base de datos y el Control que es el intermediario entre la vista y el modelo lo brinda el framework.

8.3.1 Vista. En una aplicación desarrollada con DBForms, la porción correspondiente a la vista se construye generalmente usando la tecnología JSP. Los archivos JPS pueden contener elementos estáticos de HTML, así como también elementos dinámicos que contienen código Java como definiciones o expresiones. Una de las ventajas del uso de DBForms es que posee una serie de tag libraries, las cuales encapsulan código sofisticado de Java dentro de tags de fácil manejo. DBforms hace uso del gran potencial que ofrecen las etiquetas que permiten manipular fácilmente la información contenida dentro de la base de datos.

En BDForms las JSP tienen una configuración específica que se muestra a continuación:

Figura 16. Estructura general de un formulario en DBForms – Vista



Cada JSP en DBForms puede tener una o más *tags* de tipo *form*. Cada una de ellas debe contener exactamente una tag de tipo *header*, una de tipo *body* y uno tipo *footer*, estrictamente en ese orden.

Cada una de esas tags puede contener otros elementos como campos de texto, campos para la entrada de datos, botones de acción y HTML plano y código JSP.

Las tags header and footer generalmente son empleadas para adicionar títulos a las páginas, localizar botones de navegación o de acción, campos para la entrada de textos, etc.

La tag body es usada para mostrar columnas de datos que se encuentran almacenados en la base de datos y para darle al usuario la funcionalidad de editar los datos.

8.3.2 Modelo. La principal finalidad de DBForms es la realización de operaciones en la base de datos. La definición del modelo, es decir, las tablas de la base de datos a las cuales va a acceder una aplicación realizada en DBFoms deben estar declaradas en un archivo de configuración XML llamado dbforms-config.xml, el cual es analizado y evaluado una vez la aplicación Web es iniciada.

Para la aplicación de capacitación, la definición del modelo representado en el archivo XML sería:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<dbforms-config>

       <date-format>dd-MMM-yyyy</date-format>

       <table name="INSTITUTIONS"
                    defaultVisibleFields="COUNTRY,SHORT_NAME,SITE"
                    dataAccessClass="dataaccess.DataSourceJDBCWithRowCount">
             <field name="LONG_NAME" fieldType="varchar2" size="150"/>
             <field name="SHORT_NAME" fieldType="varchar2" size="50"
isKey="true"/>
             <field name="ACRONYM" fieldType="varchar2" size="20"/>
             <field name="ADDRESS" fieldType="varchar2" size="150"/>
             <field name="CITY" fieldType="varchar2" size="39"/>
             <field name="STATE" fieldType="varchar2" size="60"/>
             <field name="ZIP_CODE" fieldType="varchar2" size="10"/>
             <field name="COUNTRY" fieldType="varchar2" size="60" isKey="true"/>
             <field name="REGION" fieldType="number" size="1"/>
             <field name="FAX" fieldType="varchar2" size="80"/>
             <field name="TELEX" fieldType="varchar2" size="80"/>
             <field name="EMAIL" fieldType="varchar2" size="80"/>
             <field name="TELEPHONE" fieldType="varchar2" size="80"/>
             <field name="P_O_BOX" fieldType="varchar2" size="26"/>
             <field name="TYPE_INST" fieldType="varchar2" size="80"/>
             <field name="FIELD_ACT" fieldType="varchar2" size="80"/>
             <field name="SITE" fieldType="varchar2" size="39" isKey="true"/>
             <field name="FECHA" fieldType="date" size="7"/>
             <field name="ACTUALIZO" fieldType="varchar2" size="20"/>
             <field name="DOMAIN" fieldType="varchar2" size="20"/>
             <field name="SCOPE" fieldType="varchar2" size="20"/>
             <field name="INSIZE" fieldType="varchar2" size="10"/>
             <field name="CREATED_BY" fieldType="varchar2" size="15"/>
             <field name="CREATED_ON" fieldType="date" size="7"/>
             <field name="COUNTRY_EN" fieldType="varchar2" size="35"/>

             <foreign-key  name="SYS_C0012815"
                   foreignTable="COUNTRIES"
                   displayType="select">
                  <reference local="COUNTRY"
                             foreign="COUNTRY"/>
             </foreign-key>

<!-- add "granted-privileges" element for security constraints -->

       </table>

       <table name="TRAINING"
              defaultVisibleFields="CODE,INIT_DATE"
              dataAccessClass="dataaccess.DataSourceJDBCWithRowCount">
             <field name="CODE" fieldType="number" size="8" isKey="true"/>
             <field name="INIT_DATE" fieldType="date" size="7" isKey="true"/>
             <field name="END_DATE" fieldType="date" size="7"/>
             <field name="TYPE_TRAINING" fieldType="varchar2" size="20"/>
             <field name="COURSE" fieldType="varchar2" size="500"/>
```

```
        <field name="DISCIPLINE" fieldType="varchar2" size="200"/>
        <field name="TRAVEL_FUNDING" fieldType="varchar2" size="150"/>
        <field name="STAY_FUNDING" fieldType="varchar2" size="250"/>
        <field name="TRAVEL_BUDGET" fieldType="varchar2" size="8"/>
        <field name="STAY_BUDGET" fieldType="varchar2" size="20"/>
        <field name="LIFE_INSURANCE" fieldType="varchar2" size="1"/>
        <field name="EVENT" fieldType="varchar2" size="6"/>
        <field name="TIMES_TRAINED" fieldType="number" size="2"/>
        <field name="SUPERVISOR" fieldType="varchar2" size="50"/>
        <field name="COUNTRY_ORIG" fieldType="varchar2" size="60"/>
        <field name="INSTITUTION" fieldType="varchar2" size="150"/>
        <field name="COUNTRY" fieldType="varchar2" size="25"/>
        <field name="REGION" fieldType="varchar2" size="1"/>
        <field name="DEPENDENT" fieldType="number" size="2"/>
        <field name="MONTHS" fieldType="number" size="6"/>
        <field name="INSURANCE_BUDGET" fieldType="varchar2" size="8"/>
        <field name="TRAVEL_COST" fieldType="number" size="9"/>
        <field name="STAY_COST" fieldType="number" size="9"/>
        <field name="COMMODITY" fieldType="varchar2" size="200"/>
        <field name="CREATED_BY" fieldType="varchar2" size="15"/>
        <field name="CREATED_ON" fieldType="date" size="7"/>
        <field name="SITE_OF_TRAINING" fieldType="varchar2" size="60"/>

        <foreign-key  name="SYS_C0012813"
              foreignTable="CLASSES"
              displayType="select">
             <reference local="TYPE_TRAINING"
                        foreign="CLASS"/>
        </foreign-key>

        <foreign-key name="SYS_C0012834"
              foreignTable="SUPERVISORS"
              displayType="select">
             <reference local="SUPERVISOR"
                        foreign="SUPERVISOR"/>
        </foreign-key>

        <foreign-key name="SYS_C0012839"
              foreignTable="ALTINST"
              displayType="none">
             <reference local="COUNTRY"
                        foreign="COUNTRY"/>
             <reference local="INSTITUTION"
                        foreign="INSTITUTION"/>
        </foreign-key>

        <foreign-key name="SYS_C0012835"
              foreignTable="DISCIPLINES"
              displayType="select">
             <reference local="DISCIPLINE"
                        foreign="DISCIPLINE"/>
        </foreign-key>

        <foreign-key  name="SYS_C0012814"
              foreignTable="USERS"
              displayType="select">
             <reference local="CODE"
                        foreign="CODE"/>
        </foreign-key>

    </table>
```

```
<table name="USERS"
       defaultVisibleFields="CODE"
       dataAccessClass="dataaccess.DataSourceJDBCWithRowCount">
      <field name="CODE" fieldType="number" size="8" isKey="true"/>
      <field name="INSTITUTION" fieldType="varchar2" size="50"/>
      <field name="F_NAME" fieldType="varchar2" size="22"/>
      <field name="L_NAME" fieldType="varchar2" size="22"/>
      <field name="GENDER" fieldType="varchar2" size="1"/>
      <field name="MARITAL_STA" fieldType="varchar2" size="10"/>
      <field name="DATE_BIRTH" fieldType="number" size="5"/>
      <field name="STATE" fieldType="varchar2" size="60"/>
      <field name="CITY" fieldType="varchar2" size="39"/>
      <field name="COUNTRY" fieldType="varchar2" size="60"/>
      <field name="REGION" fieldType="number" size="1"/>
      <field name="ADDRESS" fieldType="varchar2" size="150"/>
      <field name="SECTION" fieldType="varchar2" size="39"/>
      <field name="ACADEMIC_DEG" fieldType="varchar2" size="150"/>
      <field name="TELEPHONE" fieldType="varchar2" size="80"/>
      <field name="P_O_BOX" fieldType="varchar2" size="26"/>
      <field name="ZIP_CODE" fieldType="varchar2" size="10"/>
      <field name="FAX" fieldType="varchar2" size="80"/>
      <field name="ENTRY_DATE" fieldType="date" size="7"/>
      <field name="UPDATE_DATE" fieldType="date" size="7"/>
      <field name="REFERENCE" fieldType="varchar2" size="45"/>
      <field name="USER_INST" fieldType="number" size="2"/>
      <field name="CHARGE" fieldType="varchar2" size="60"/>
      <field name="FIELD_ACT" fieldType="varchar2" size="40"/>
      <field name="NETWORK" fieldType="varchar2" size="3"/>
      <field name="COUNTRY_I" fieldType="varchar2" size="60"/>
      <field name="INSTITUTION_2" fieldType="varchar2" size="39"/>
      <field name="POST_GRADUATE" fieldType="varchar2" size="80"/>
      <field name="ACRONYM" fieldType="varchar2" size="20"/>
      <field name="DOCUMENT" fieldType="varchar2" size="80"/>
      <field name="USER_MOD" fieldType="varchar2" size="2"/>
      <field name="EMAIL_ADDRESS" fieldType="varchar2" size="100"/>
      <field name="CREATED_BY" fieldType="varchar2" size="15"/>
      <field name="CREATED_ON" fieldType="date" size="7"/>
      <calc name="ROW_NUM" fieldType="int"/>

      <foreign-key  name="SYS_C0012810"
            foreignTable="COUNTRIES"
            displayType="select">
           <reference local="COUNTRY"
                    foreign="COUNTRY"/>
      </foreign-key>

      <foreign-key  name="SYS_C0012811"
            foreignTable="INSTITUTIONS"
            displayType="none">
           <reference local="COUNTRY_I"
                    foreign="COUNTRY"/>
           <reference local="INSTITUTION"
                    foreign="SHORT_NAME"/>
           <reference local="INSTITUTION_2"
                    foreign="SITE"/>
      </foreign-key>

      <foreign-key  name="SYS_C0012809"
            foreignTable="TIPOUSUA"
            displayType="select">
           <reference local="USER_INST"
```

```
                        foreign="CODE"/>
            </foreign-key>

<!-- add "granted-privileges" element for security constraints -->

      </table>


      <!-- ========== Connection ============================== -->

      <dbconnection
            name   = "jdbc:oracle:thin:@SERVIDOR:PUERTO:INSTANCIA"
            isJndi = "false"
            conClass  = "oracle.jdbc.driver.OracleDriver"
            username = "xxxxxxx"
            password  = "xxxxxxx"
      />

</dbforms-config>
```

8.3.2.1 Definición del acceso físico a la base de datos. DBForms debe estar en la capacidad de crear conexiones a la base de datos, la cual contiene las tablas (vistas) y campos declarados en el archivo *dbforms-config.xml*.

En este caso en la aplicación no se empleó un pool de conexiones, sino que se definió un simple acceso a la base de datos como se muestra:

```
      <dbconnection
            name   = "jdbc:oracle:thin:@SERVIDOR:PUERTO:INSTANCIA"
            isJndi = "false"
            conClass  = "oracle.jdbc.driver.OracleDriver"
            username = "xxxxxxx"
            password  = "xxxxxxx"
      />
```

8.3.3 Controlador. Empleando DBForms el desarrollador no tiene la necesidad de generar el código que maneja la lógica de la aplicación, la herramienta lo provee. El controlador posee varios componentes:

- **ControllerServlet:** Este servlet es el único punto de entrada para todas las peticiones HTTP entrantes realizadas por los clientes.

- **EventEngine:** representa un “ayudante” del ControllerServlet, que se encarga de filtrar las peticiones HTTP a WebEvents e instanciarlos.

- **WebEvent Objects**: Todos los objetos derivados de la super-clase abstracta WebEvent tienen la propiedad de inicializarse a sí mismos leyendo una determinada petición HTTP. Estos eventos pueden ser ejecutados directamente por el Controlador o por la Vista.

La manera como se ejecuta una acción realizada por el usuario se define con los siguientes pasos:

- El usuario presiona el botón “borrar registro” de su aplicación.

- El navegador envía vía HTTP-POST los datos al *ControllerServlet*.

- El *ControllerServlet* delega la petición entrante al *EventEngine*, el cual determina el evento principal (el usuario accionó explícitamente el acontecimiento al presionar el botón).

- El *EventEngine* analiza la petición y determina la clase de acción que el usuario quiere que se ejecute.

- Luego crea el *WebEvent* apropiado, en este caso *DeleteEvent* y delega el request Object a este *WebEvent* creado recientemente. Posteriormente el *Eventengine* retorna el evento recién creado y al *ControllerServlet*.

- Si el evento es un *DatabaseEvent* el *ControllerServlet* le dice que ejecute la operación; otros eventos como NevigationEvent, etc, son delegados al componente de la Vista (View) apropiado.

- El ControllerServlet invoca el EventEngine nuevamente para verificar si hay más eventos que van a ser ejecutados. Si es así, los Objetos tipo WebEvent son creados y ejecutados.

- El ControllerServlet determina el componente de la Vista (View) al cual la petición debe ser remitida. Si lo encuentra, el ControllerServlet invoca el componente y remite la petición.

- Si el componente de la Vista es una página JSP que contiene tags de DBForms, esas tags buscarán eventos de navegación para ejecutar y finalmente se generará la respuesta al usuario.

- La respuesta es mostrada al usuario en el navegador Web.

9. PRUEBAS – IMPLEMENTACION

Dentro de la metodología de desarrollo empleada en la Unidad de Sistemas de Información, USI, del CIAT, el plan de pruebas ocupa un lugar destacado, pues del resultado de éstas depende en gran parte la satisfacción del usuario con el producto final.

El tipo de pruebas mas empleadas en la USI son las de caja blanca, aunque durante el proceso de desarrollo el programador va realizando pruebas a medida que va elaborando los casos de uso, con el fin de que al momento de la realización de las de caja blanca, el número de no conformidades encontradas sea el mínimo. Por otro lado, para obtener retroalimentación de parte de los usuarios, se les brinda acceso a la aplicación la cual se instala en un servidor de pruebas.

En este caso específico, las pruebas se hicieron sobre cada módulo del sistema, es decir, Users, Training, Institutions, Countries, Continents, Classes, Disciplines, Budgets, Supervisors y Usuarios de la Aplicación.

Para cada uno de los módulos que componen el sistema se verificaron los siguientes puntos, que se pueden clasificar en Seguridad y Funcionalidad de la aplicación:

9.1 PRUEBAS DE SEGURIDAD Y ACCESO AL SISTEMA

- Que solo los usuarios creados en la base de datos tuvieran acceso a la aplicación.

- Que según el perfil del usuario se desplegara el menú con las opciones correspondientes.

- Que al finalizar el tiempo de la sesión no se pudiera acceder a ninguna de las opciones del sistema.

- Que al digitar en el URL la dirección de alguna de las páginas de la aplicación, sin estar registrado, no se pudiera tener acceso a ella.

9.2 PRUEBAS DE FUNCIONALIDAD DEL SISTEMA

- Ingreso de valores que no corresponden al tipo de dato especificado para cada uno de los campos:

    - Numérico
    - Fecha
    - Caracteres

- Ingreso de valores de mayor tamaño que el especificado en la base de datos.

- Almacenamiento de los valores ingresados en cada uno de los formularios confrontándolos en la base de datos.

- Actualización correcta de los datos modificados, confrontándolos igualmente con la base de datos.

- Que los resultados de las búsquedas correspondieran al/los término(s) ingresados según el tipo seleccionado, es decir, que contuviera el término o que fuera exacto.

Como la aplicación se encontraba en funcionamiento bajo el esquema cliente – servidor, no era posible tener acceso a la base de datos sobre la cual funciona, durante el proceso de desarrollo para conservar la integridad de los datos, por lo que se hizo necesaria la creación de un esquema en un servidor de base de datos de prueba.

La base de datos mencionada fue poblada con una cantidad mínima de datos, en promedio 50 registros por tabla, y con ellos se hicieron las pruebas durante el desarrollo y las 2 versiones que se realizaron para los usuarios finales.

La finalización del proceso de codificación coincidió con el cambio del servidor de base de datos de producción por uno de mayor capacidad, dejando el anterior servidor disponible para la realización de las pruebas de caja blanca.

En ese momento, contando con una base de datos de pruebas que contenía la totalidad de los registros almacenados desde el año 1970 aproximadamente, se procedió a realizar las pruebas, el resultado no pudo ser peor: trabajando bajo los estándares de DBForms, es decir, la manera que este conjunto de *tags* emplea para obtener registros o acceder a ellos, el servidor o contenedor de servlets, en este caso Tomcat 4.1.3, literalmente se caía por falta de memoria… no soportaba el manejo de tal cantidad de registros, que en la tabla mas grande era de casi 20.000.

El problema se presentaba en la "entrada" a cada una de las opciones del menú, en donde se desplegaba la lista paginada de los registros de la base de datos, al parecer éste no es uno de los fuertes de DBForms, pues al tratar de acceder al listado, el servidor se caía de inmediato.

Por otro lado, se presentaban problemas al momento de la creación de un ítem de alguna de las entidades, precisamente aquellas que en la base de datos contenían gran número de registros (mas de 9000): Users, Training, Institutions, donde los tiempos para el despliegue de los formularios podía superar el minuto y al momento de actualizar o guardar el formulario el servidor se caía, nuevamente por falta de memoria.

Ante esta situación y teniendo en cuenta que la fecha de entrega ya se aproximaba, se debió hablar con los usuarios y explicarles la situación, con lo cual se ganó algo de tiempo, durante el cual se plantearon diversas opciones: se consultó con otros desarrolladores de Unidad de Sistemas, pues dos de las aplicaciones más grandes creadas en el CIAT, están desarrolladas en DBForms, pero no se pudieron aclarar los interrogantes presentados ante esta situación, pues el manejo de datos en estas aplicaciones no superaban los 1000 registros en la tabla mas grande. Una de las opciones que se presentó ante la falta de información sobre el problema, fue la de diseñar la parte afectada en Struts empleando Hibernate, lo cual implicaba cambiar la arquitectura de la aplicación y mas tiempo en el desarrollo, pues era necesario el aprendizaje de estas tecnologías.

Se decidió primero intentar modificar la aplicación, para lograr que funcionara sin tumbar el servidor: para esto se eliminó el despliegue de los listados de Users,

Trainings e Institutions en las jsp iniciales, y solo se dejó disponible en ellas la opción de búsqueda para el acceso a los registros de la base de datos. El otro cambio que se realizó fue en la manera en la que DBForms permite el acceso a un registro: en lugar de hacerlo de la manera tradicional, se decidió emplear funciones de JavaScript.

Con DBForms para actualizar una institución se empleaba la siguiente sentencia:

```
<db:gotoButton
   destination="/INSTITUTIONS_update.jsp"
   destTable="INSTITUTIONS"
   destPos="<%= position_INSTITUTIONS %>" flavor="image"
   src="images/bot_actualizar.gif"/>
```

Con el cambio realizado se pasan los parámetros a una función de JavaScript, la cual se encarga de enviarlos vía URL (Uniform Resource Locator) a la jsp destino, para obtener el registro buscado, así:

```
<a href='javascript:Goto_Update(
      "<db:label fieldName="COUNTRY"/>",
      "<db:label fieldName="SHORT_NAME"/>",
      "<db:label fieldName="SITE"/>" )'>
      <img src='images/bot_actualizar.gif' border='0' alt='UPDATE'>
</a>

function Goto_Update(country,short_name,site){
      short_name=ReplaceAll(short_name,'&','%26');
      document.dbform.action="INSTITUTIONS_update.jsp?country="
      +country +"&short_name=" + short_name + "&site=" + site;
      document.dbform.submit();
}
```

De esta manera se logró disminuir el tiempo de respuesta en aproximadamente el 80%, y se logró que el servidor no se cayera al realizar las consultas sobre las tablas con más de 9000 registros.

Las demás entidades no presentaron problemas y se dejaron con el funcionamiento tradicional de DBForms, por cuestiones de tiempo y porque se estimó que el crecimiento de las mismas sería mínimo.

Otro aspecto a destacar fue el cambio de la plataforma de funcionamiento para todas las aplicaciones desarrolladas en la USI, como se muestra a continuación:

Servidor de Aplicaciones / Contenedor de Servlets	
- Tomcat 4.1.3	- Tomcat 5.0.28
Versión de Java	
- Java 1.4.2	- Java 1.4.2_09

Esto se debió al cambio del servidor empleado hasta el momento por uno más robusto.

Con el cambio de plataforma se presentó la migración de la Aplicación de Capacitación. Uno de los principales inconvenientes presentados, fue el hecho de que DBForms presenta problemas con los *commons* de apache por lo que cual fue necesario bajar la nuevas versiones de ellos y copiarlos en la carpeta *lib* del proyecto, teniendo en cuenta borrar los *jar* que tuvieran nombre diferente.

commons-beanutils.jar
commons-collections-3.1.jar
commons-digester-1.7.jar
commons-logging.jar
commons-logging-api.jar
log4j.properties
log4j-1.2.9.jar

Otro problema se presentó con las versiones del *common-logging* y *log4j*. Para solucionar fue necesario modificar la configuración de archivo Catalina de Tomcat así:

- Editar el conf/catalina.properties y adicionarle a la propiedad common.loader la ruta: ${catalina.home}/shared/lib/*.jar

- Copiar log4j-1.2.9.jar and commons-logging.jar al directorio shared/lib de Tomcat.

- Copiar un archivo log4j.properties en el directorio common/classes de Tomcat.

La parte de correo también se vio afectada con el cambio del Servidor; para solucionar este inconveniente se bajó la última versión de la *api* JavaMail 1.3.3 y se copió en la carpeta *lib los jar mail.jar* y *activation.jar*.

Superados estos inconvenientes se realizaron nuevas pruebas a la aplicación para finalmente ser puesta en el nuevo servidor de aplicaciones para su uso.

10. INSTALACIÓN

Para el correcto funcionamiento de la aplicación es necesario contar con un servidor de aplicaciones y uno de bases de datos:

- Servidor de aplicaciones:
  - Apache Tomcat/5.0.28
  - JDK Virtual Machina: Java Runtime Environment (JRE) 1.5.0_04
  - Sistema Operativo Linux / Windows

- Servidor de bases de datos:
  - Base de datos Oracle 9i

10.1 CREACIÓN DE LA BASE DE DATOS

Para la creación de la base de datos se siguen los siguientes pasos:

- Crear el esquema en el servidor de bases de datos.

- Correr en ese esquema los scripts de creación de la base de datos.

10.2 INSTALACIÓN DE LA APLICACIÓN EN EL SERVIDOR DE APLICACIONES

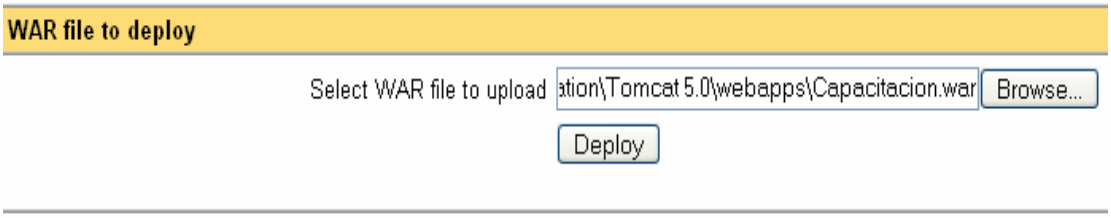
- La aplicación se entrega como un archivo WAR, el cual es un archivo comprimido, al igual que un JAR, que contiene todos los archivos necesarios para la aplicación Web.

- Este fichero debe copiarse en el directorio único que hay que hacer es copiar este fichero WAR al directorio %TOMCAT_HOME%\webapps, luego será Tomcat el que se encargue de descomprimir el archivo cuando se arranque el servidor.

- **Nota:** %TOMCAT_HOME% indica el directorio donde se encuentra instalado Tomcat.

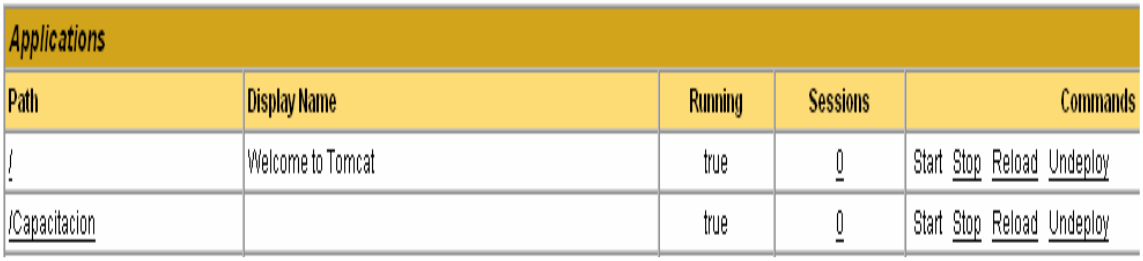
- Otra manera de instalar la aplicación en el servidor es desde el Tomcat Manager: iniciando el servicio ingrese el password y usuario.

Figura 17. Deploy de la aplicación desde Tomcat



- En la pantalla que se muestra seleccione la ubicación del archivo WAR y a continuación presione el botón Deploy, con esto la aplicación quedará lista para su funcionamiento.

Figura 18. Aplicación después de realizado el deploy



11. CONCLUSIONES

- Durante la realización del proyecto se aplicó la metodología de desarrollo de la Unidad de Sistemas de Información del CIAT, USI, la cual “se ha basado en la metodología de Extreme Programming (XP) de Kent Beck, desarrollo de Procesos y se ha incluido Ideas de otros modelos como UML y Teoría de Gerencia de Proyectos y Experiencias CIAT”.

- La metodología de desarrollo incluye las etapas de Planeación, Análisis, Diseño, Codificación, Pruebas e Implementación, las cuales al ser cumplidas en su totalidad garantizó el éxito del proyecto y en consecuencia la satisfacción de los usuarios.

- Uno de los principales inconvenientes encontrados durante el desarrollo de la aplicación radicó en la relación con los usuarios que en la mayoría de las ocasiones no estaban conscientes de las verdaderas necesidades que se requerían en el software y solicitaban características o funcionalidades que no habían sido consideradas en la propuesta de desarrollo y que por lo tanto no fueron tenidas en cuenta.

- DBForms, el framework empleado para el desarrollo de la aplicación cumplió en gran parte con los requerimientos especificados, las fallas que se presentaron tuvieron relación con el volumen de datos manejados en la base de datos, lo cual se solucionó implementando el llamado a páginas mediante el uso de javascripts en lugar del método implementado oficialmente por DBForms.

- Para aplicaciones futuras que requieran mayor grado de personalización o consultas más complejas, es recomendable el uso de otras tecnologías como Struts, Java Server Faces JSF, Spring, entre otras, con la ayuda de un mapeador objeto-relacional como Hibernate.

- Con la realización de este proyecto se pudo participar por primera vez en el desarrollo de una aplicación Web, donde se empleó software libre en su mayoría, como el lenguaje de programación Java, el contenedor de servlets Tomcat, el servidor de aplicaciones Apache y la plataforma de desarrollo Eclipse, las cuales son en la actualidad de gran uso en ambientes empresariales.

BIBLIOGRAFÍA

COLUNGE, Doryan; HACEY, Diego. Metodología de Desarrollo de Software USI-CIAT. Versión 2. Santiago de Cali, 2005. 16 p.

PEER, Joachim. DBForms User's Guide Version 2.5 [en línea]. Viena: DbForms Source Forge, 2005. [consultado 10 de Mayo, 2005]. Disponible en Internet: http://jdbforms.sourceforge.net/UsersGuide/pdf/DbForms_UsersGuide.pdf

Wikipedia: la enciclopedia libre [en linea] Florida: Wikimedia Foundation 2006. [consultado 15 de Junio, 2005]. Disponible en Internet: http://es.wikipedia.org/wiki/portada

Windows Services. Introduction to data Modeling [en línea]. Austin : Information Technology Services at The University of Texas at Austin. 1994. [consultado 03 de Mayo, 2005] Disponible en Internet: http://www.utexas.edu/its/windows/database/datamodeling/dm/erintro.html

Anexo A. Datos a ingresar para los casos de uso

Create New User

- **Personal Information**
    - Entry_Date
    - User Code*
    - Last Name
    - First Name
    - Date of birth
    - Country of Origen*
    - Gender (M/F)
    - Marital Status
    - Address
    - City
    - State
    - Email
    - Fax Number
    - Telephone
    - Document Number
    - Academic Degree
    - Graduate Degree
    - Type of User
    - Name Person Updating

- **Institucional Information**
    - Country
    - Name Of Institution
    - State
    - City
    - Acronym
    - Address
    - Present Position
    - Field Of Activity
    - Section/Department
    - Referente

Create New Training

- User Code
- Initiation Date
- Termination Date
- Type of Training
- Site Of Training
- Course
- Project
- Discipline
- Travel Funding
- Stipend Funding
- Times trained
- Supervisor
- Candidate Country of Origen
- Country of Institution
- Name Of Institution
- Insurance budget
- Life Insurance

Create New Event

- Name
- Initiation Date
- End Date
- Country
- City
- Number of Participants
- Coordinator

Create New Institution

- Name of Institution
- Short Name of Institution
- Acronym
- Address
- Site of Institution*
- City
- State
- Country*
- Telephone
- Fax Number
- E-mail

- Main Activity
- Name Person Updating
- Entry Date

Create New AltInstitution

- Name of Institution
- Country

Create New Country

- Country (español)
- Country (Inglés)
- Continent
- Region
- Status (Enabled/Disabled)

Create New Continent

- Continent Name

Create New Discipline

- Disciplina (Inglés)
- Disciplina (Español)
- Type (I=Discipline / R=Commodity)
- Status

Create New Class

- Class (Español)
- Class (Inglés)
- Status (Disabled – Enabled)

Create New Buget

- Budget
- Name

Create New Supervisor

- Supervisor Name

Create New User App

- Name
- Login
- Pasword
- Email
- Rol (Outposted/Adminsitrator)

Anexo B. Paper

DESARROLLO DE UNA APLICACIÓN WEB PARA LA UNIDAD DE INFORMACION Y FORTALECIMIENTO DE CAPACIDADES DEL CIAT

Lida del Rosario De la Hoz Gómez

Universidad Autónoma de Occidente, Km 2 vía Jamundí,
ldelahozg@gmail.com, Santiago de Cali

Abstract: En el presente documento se exponen aspectos relacionados con la metodología de desarrollo de software que se empleó durante la realización de la "Aplicación Web para la Unidad de Información y Fortalecimiento de Capacidades del CIAT", destacando aspectos del diseño y de la arquitectura seleccionados, así como las características del framework empleado para la implementación.

Keywords: Software engineering, analysis, databases, architectures, information, software, programming, applications, testing, diagrams.

1. INTRODUCCIÓN

El proyecto "Desarrollo de una Aplicación Web para la Unidad de Información y Fortalecimiento de Capacidades del CIAT" se desarrolló en la Unidad de Sistemas de Información del Centro Internacional de Agricultura Tropical bajo la coordinación del Ingeniero Doryan Colunge, con el fin de cambiar la arquitectura cliente-servidor existente por la arquitectura web, que contrario a la primera, permitiera el acceso a la base de datos de INFORCAP no solo al personal localizado en la sede de Palmira, sino también al que se encuentra en las estaciones outposted distriubuidas en Africa, Asia y Centroamérica.

Este documento describe la metodología con la cual se desarrolló el proyecto, las

decisiones que se tomaron y el porqué, con el fin de que el lector conozca de manera general, los pasos que se siguieron para la realización del mismo.

2. METODOLOGIA

La metodología de desarrollo de software que se empleó para el desarrollo del proyecto es propia de la Unidad de Sistemas de Información del CIAT y recoge las mejores prácticas de otras metodologías como la Extreme Programing, Desarrollo de Procesos, y la Teoría de Gerencia de Proyectos, con la ayuda del lenguaje de modelado unificado UML.

La metodología consta de las siguientes etapas: Planeación, Análisis, Diseño, Codificación, Pruebas e Implementación, a continuación se describe lo que se realizó en cada una de ellas.

2.1 Subproceso de Planeación.

Una vez se tuvieron las necesidades básicas de los usuarios y fueron clasificadas para la realización de un nuevo proyecto, éstas se entregaron al grupo de trabajo correspondiente, líder y analista; el equipo se reunió con los usuarios para realizar el levantamiento inicial de requerimientos y se generaron las historias preliminares del proyecto.

Las historias se analizaron y con ello se realizó la propuesta inicial, donde se incluyó una estimación de los costos, los recursos necesarios y el cronograma inicial para el desarrollo del proyecto.

2.2 Análisis.

En esta etapa las historias fueron revisadas nuevamente y con la ayuda de los usuarios se obtuvo el listado de requerimientos con un buen nivel de detalle, los cuales fueron presentados ante los usuarios para que dieran su aprobación.

Con el listado completo de requerimientos se realizaron los diagramas de casos de uso y el detalle de cada uno de ellos.

2.3 Diseño.

En esta etapa se realizaron las modificaciones al modelo entidad relación según los requerimientos de los usuarios, además se definió que se emplearía el framework DBForms para la codificación del proyecto, pues facilitaría mucho la labor de desarrollo, ya que es una herramienta para desarrollo rápido que implementa las especificaciones Java Servlets y Java Server Pages, empleadas en la Unidad de Sistemas, además de que implementa el patrón de diseño Modelo – Vista – Controlador que era el que se deseaba implementar en el proyecto.

No se realizaron diagramas de clase ni de secuencia puesto que al emplear DBForms no se crean clases, el framework emplea un Controlador que se encarga de manejar la lógica de la aplicación.

Una vez establecido lo anterior, se procedió a crear un repositorio para la aplicación en el CVSNT que es el software que permite el control de versiones, es decir, lleva el control de los cambios que se hacen en la aplicación durante su desarrollo.

Se realizaron también la matriz de requerimientos de prueba y los casos de prueba del sistema.

2.4 Codificación.

Durante esta etapa se realizó la programación en lenguaje Java de las funcionalidades de la aplicación según los casos de uso realizados, en la plataforma de desarrollo Eclipse con el plugin MyEclipse que le confiere las capacidades para trabajar bajo la plataforma de Java Enterprise Edition, J2EE.

Para la creación de la aplicación con DBforms se empleó la herramienta basada en SWING DevGui, con la cual se creó el archivo dbforms-congif.xml, uno de los principales dentro de la aplicación, y que contiene información sobre cada una de las tablas de la base de datos, sus atributos y también posee la definición del acceso físico a la base de datos; la herramienta también permitió la creación de los archivos JSP o las "vistas", las cuales permiten que los usuarios interactúen con la base de datos. Estas JSP's pueden ser de ingreso, actualización y listado de la información contenida en la base de datos. Durante la codificación estos archivos fueron reemplazados por completo, se le adicionaron scripts realizados en JavaScript en los casos en los que fue necesario y en la parte de validación se empleó el framework que acompaña por defecto a DBForms, el Apache's Commons-Validator, el cual permite realizar validaciones del lado del servidor como verificar obligatoriedad y tipo de campos, al igual que del lado del cliente mediante el uso de métodos de JavaScripts.

2.5 Pruebas.

Durante esta etapa con la ayuda de la matriz de requerimientos de prueba se llevaron a cabo las pruebas necesarias con servidores de aplicaciones y de datos de prueba, con las cuales se determinaron las no conformidades existentes y se hicieron los cambios necesarios al software para que cumpliera con los requisitos de los usuarios.

2.6 Implementación.

Para la implementación se generó un archivo *war* de la aplicación y se instaló en el servidor de aplicaciones Tomcat, con la base de datos de producción Oracle y se dejó a disposición de los usuarios.

3. ARQUITECTURA

Se implementó la arquitectura de tres capas, donde el cliente es el navegador web (Mozilla, Internet Explorer), el middleware lo representan las vistas (es decir las jsp) y el controlador, que lo provee DBForms, y la capa de datos lo representa el modelo (la base de datos – archivo xml de configuración de la aplicación).

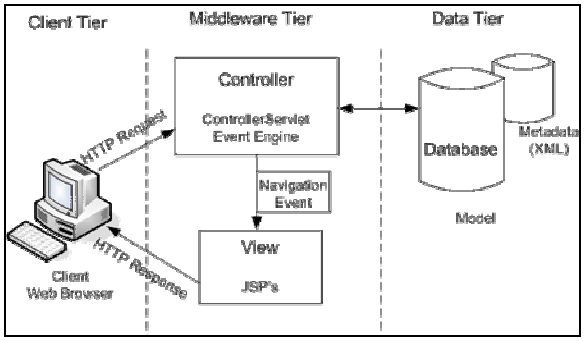


Fig. 1. Diagrama de la arquitectura del sistema.

La siguiente figura se muestra la distribución del procesamiento entre los distintos equipos que conforman la solución, incluyendo los servicios y procesos de base.

El acceso a la aplicación se realiza a través de un browser que puede ser Internet Explorer, Nestcape o Firefox. En la parte de los servidores se cuenta con Apache como servidor Web haciendo peticiones a Tomcat como contenedor Java, que se encarga de responder peticiones de paginas jsp y de java (.java); éstos hacen uso de archivos de configuración y de mapeo en XML.

Los archivos XML contienen la configuración del mapeo de tablas de bases de datos relacionales. Estas bases de datos mostradas en el tercer nivel pueden ser de motores como: Oracle, SQL Server, Postgresql o MySQL, los cuales cuentan con los esquemas y las tablas necesarias para el almacenamiento de datos de aplicación.

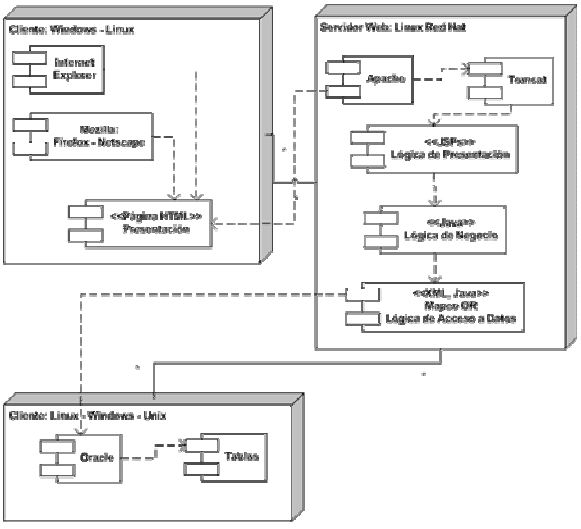


Fig. 2. Diagrama de componentes del sistema.

4. SOBRE EL USO DE DBForms

La principal razón por la que se escogió DBForms como base para el desarrollo de la aplicación, fue su facilidad de uso y el hecho que permite desarrollar aplicaciones en corto tiempo.

A pesar de estas características y de que permite el uso de validaciones, ordenamientos, búsquedas entre otras, el framework se queda corto si lo que se desea es personalizar la aplicación, pues las principales funcionalidades como la inserción, actualización, listado y eliminación de la información de la base de datos vienen prediseñadas lo que imposibilita tener el control total de la aplicación, además de que para el manejo de una gran cantidad de registros de la base de datos puede causar que el servidor consuma mas memoria que la establecida y la aplicación se caiga.

Se recomendaría este framework para aplicaciones que no requieren características especiales adicionales a la inserción, actualización y listado de tablas de la base de datos y que no manejen altos volúmenes de registros.

5. CONCLUSIONES

- Gracias al desarrollo de la aplicación se logró que el personal de las estaciones outposted del CIAT pudieran incluir su propia información en la base de datos, evitando el proceso a través de intermediarios lo cual resultaba poco eficiente y se reflejaba en la actualización tardía de los datos.

- El uso de la metodología de desarrollo propia de la Unidad de Sistemas de Información permitió en gran medida el cumplimiento de las fechas de culminación especificadas para cada etapa, así como también permitió tener mayor control sobre cada una de las actividades involucradas durante el desarrollo de la aplicación.

- Emplear DBForms para el desarrollo de una aplicación es recomendable cuando la mayor parte de las operaciones a realizar con la aplicación se refieran a inserciones, consultas y actualizaciones de datos, si la aplicación es compleja se recomienda el uso de otros frameworks como Struts, Java Server Faces, Spring, entre otros y para el manejo de las operaciones relacionadas con la base de datos Hibernate, un mapeador objeto relacional (ORM) que ayuda evitar la diferencia entre el paradigma orientado a objetos del lenguaje Java y las base de datos relacionales, que no manejan objetos.

REFERENCIAS

Proceso de desarrollo de software para los requerimientos del usuario. CIAT. 2005

http://jdbforms.sourceforge.net/index.html
DBForms HomePage

http://www.uml.org/, The Unified Modeling Language – UML.